This is to certify that the
thesis entitled

GRAPH-BASED EMAIL PRIORITIZATION

presented by

Ronald Nussbaum

has been accepted towards fulfillment
of the requirements for the

M.S. degree in Computer Science

Major Professor's Signature

8/18/2008

Date

Master's Thesis

GRAPH-BASED EMAIL PRIORITIZATION

By

Ronald Nussbaum

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Computer Science

Computer Science and Engineering

2008

ABSTRACT

GRAPH-BASED EMAIL PRIORITIZATION

By

Ronald Nussbaum

The exponential growth of the Internet over the last two decades has raised a number of issues. Unsolicited bulk email, or spam, has become a huge burden on individuals and businesses alike. Typically sent out in mass quantities, many approaches have been taken to fight email spam. From simple text-based filters, to whitelists and blacklists, and increasingly complex Bayesian learners, these approaches have met with varying degrees of success.

The ubiquity of email presents a second problem. An individual may receive tens or hundreds of legitimate email messages per day. Whether this excess email is legitimate or not is unimportant once it becomes an unreasonable burden on ones time. It must still be filtered, or better yet ranked, so that valuable time is not wasted.

This thesis uses graph-based methods to prioritize incoming email messages. A model is first constructed from the header information of previously received messages. The model is then used to predict which email messages in a user's inbox are most likely to be urgent and in need of a response. Once ranked, a user may read as many of the higher priority messages as time permits. Lower priority messages are ignored or saved until later.

In the first part, a model is created for each user solely from that user's email history. In the second part, the model for each user incorporates the email histories of other users as well. Results are generated from tests using the Enron email dataset.

To My Grandfather

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Dr. Abdol-Hossein Esfahanian, Dr. Pang-Ning Tan, and Dr. Eric Torng for serving as my advisors and co-advisor, as well as teaching some of the best courses I have had the opportunity to take while a student at Michigan State University. Their guidance and thirst for knowledge have kept me focused and motivated. In particular, I am thankful to Dr. Esfahanian for encouraging me to apply to the Ph.D. program here.

Further thanks are due to other professors I have taken courses with: Dr. Bill Punch, Dr. Erik Goodman, Dr. Charles Ofria, Dr. Sakti Pramanik, Dr. Kurt Stirewalt, Dr. Joyce Chai, and Dr. Sandeep Kulkarni. Specifically, Translation of Programming Languages with Dr. Ofria was a great experience. Also due credit are many of my fellow graduate students: Jerry Scripps, Adam Jensen, and others too numerous to mention.

I would like to thank Linda Moore and the rest of the Computer Science and Engineering staff members. They have been invaluable resources in navigating administrative issues. I am also indebted to Dr. Peter Knupfer, Heather Hawley, Dennis Boone, and the rest of my colleagues at MATRIX for providing not only funding, but an excellent place to work and learn.

Finally, I would like to thank my friends and family for their support and patience. In particular, I am grateful to John Schoolenberg, whom I dedicate this thesis to.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Description

The motivation for email prioritization goes far beyond simply fighting spam. Even a spam filtering algorithm with perfect detection of junk email is rendered useless if the remaining quantity of incoming mail overwhelms the user. Rather, the intent is to comfortably manage an otherwise overflowing inbox. Through ranking algorithms, lower priority email messages may be dealt with so that the use of email remains a net productivity gain, rather than a burden on the user.

Once ranked, it is up to the user to decide how to use the resulting information to fit their own needs. If the user only has a short amount of time available to read their email, lower priority messages can be ignored until later. Alternately, if the volume of incoming email is overwhelming, lower priority email might not be read at all, or only given a cursory glance. Other users may have such a low volume of incoming email that the rankings are reduced to merely a spam detection algorithm. Ultimately, it is expected that each user will mix and match these methods according to their personal needs. This paper does not consider such strategies further.

1.2 Goals

The overarching goal of this thesis is to develop models that accurately prioritize incoming email messages. Rather than trying to analyze the text in the email subject and body, the focus is on making use of the header information to represent the history between each pair of people. The first stage of testing is simply to take a corpus of email, and track the history of each pair of people. From this one can take the response rates and response times between pairs of people, and use that as a basis of prediction.

The primary benefit of only using direct information about pairs of people is that a model can be built for a user without needing to know anything about the email history between any other pair of people. However, this approach is unable to provide useful results when there is little or no relationship between a pair of people. Thus, it is useful to view the relationship between two people as not just the email history between them, but rather the sum of their own correspondence, augmented with the email history of their common neighbors as well.

In the second stage of testing, a larger body of email from a single domain is used to build a model. The expectation is that this model will be significantly more accurate than one built solely from the email of a single user. As explained in the following section, the difficulty here is in finding a collection of email that is suitable as dataset.

1.3 Text Corpus

It is unreasonable to expect that the recipient of an email will have access to the email history of every potential sender to build a model from. At most, it is feasible

to assume access to email history within a single domain, or perhaps a collection of related domains. Unfortunately, most publicly available email datasets are fairly small, or oriented towards spam detection. Due to privacy issues, many are artificially generated as well.

One large, publicly available email corpus is the Enron email dataset. This is a real dataset, made public in the aftermath of the Enron financial scandal and subsequent bankruptcy. It contains a good balance of internal email versus email originating from outside the enron.com domain or being sent outside of the enron.com domain (Table 1.1). The dataset contains a reasonable, but not overwhelming, amount of spam.

Table 1.1: Distribution of email addresses by domain

Domain name	Occurrences
enron.com	27565
aol.com	3117
hotmail.com	1973
yahoo.com	1470
haas.berkely.edu	648
msn.com	490
earthlink.net	388
dynegy.com	281
williams.com	248
worldnet.att.net	247
Other	41191
Total	77618

Some background information specifically regarding the release of Enron email dataset can be found in a Salon.com article [8]. Historical [18], economic [4, 10], and ethical [27] analysis of the scandal is available in journal articles, books, magazine articles, and news sources too numerous to mention, and is not covered here. Suffice it to say, Enron Corporation was a significant American energy company which collapsed amidst financial scandal in 2001. The dataset is large, and covers nearly a two year period before the company filed for bankruptcy. Despite the size of the company, the email contained therein represents only a portion of Enron's total volume of email during this time.

Chapter 2

Problem Statement

2.1 Users

In the prediction models generated for email prioritization, a node represents an individual, while a link represents the relationship between a pair of individuals. However, nodes can represent different categories of individuals. For the purposes of this thesis, the term *user* specifically refers to an individual for whom a prediction model is being built, or another individual within the domain whose email history is known. Those outside the email domain whose email history is unknown - save for their correspondence with users inside the domain - are simply referred to as *people* or *persons*. The latter terms are also used in cases where the email history of an individual may or may not be known. In the case of the Enron dataset, those individuals in the enron.com domain whose personal email history is not included in the dataset are not referred to as users.

A user or a person is defined as a single email address. Although this may appear to be a trivial statement, a single individual may have any number of email addresses, each of which may or may not be forwarding to one of the others. In a real system, one would at least want to consolidate multiple identities that are explicitly known

via forwarding rules. However, in the case of the Enron dataset, the most common situation is where a user manually forwards the occasional email to a personal email address of theirs, such as a Hotmail account. These occurrences are simply ignored, as this would require the manual detection of such aliases, something that is relatively unfeasible given the size of the Enron dataset. That is, separate identities are not detected, and are treated as different people. Also, typos may occur when a person manually enters the addresses of recipients and secondary (carbon copy) recipients in an email. These are also not merged, due to the size of the dataset, and the fact that there is a lack of responses from incorrectly entered recipients means that these relationships should not affect results.

2.2 Spam

Trying to filter out spam, or junk email, raises a difficult question: What exactly constitutes spam? Although the definition is somewhat subjective, clearly bulk email sent from virus-infected computers hawking marginally legal products qualifies as spam. If the sender's identity is real, and the product being advertised is legal, the definition is less clear. A legitimate, if highly annoying electronic commerce site might send out regular email to every customer who has ever done business with them, or created a login at their website. Taken to the extreme, messages from mailing lists an individual is automatically subscribed to at a work or school environment could be considered spam.

This thesis takes the middle ground and defines spam to be all unsolicited bulk email. Any existing relationship with an organization, no matter how trivial, precludes any email from them as being considered spam. However, by shifting the focus away from the problem of detecting spam to the problem of prioritizing all

incoming email, there is no longer a need to agonize over special cases. Instead, the emphasis is on filtering out unwanted email. Whether this is done automatically or manually, the goal is to devise an algorithm to than can accurately prioritize email. A higher priority email is one that is more likely to be read and more specifically responded to. Another issue is deciding on a proper threshold below which to ignore remaining email. As a user should have little or no history with the sender of anything that might remotely qualify as a junk email, in particular a lack of reciprocal messages, the exact definition of the term is not as important as it would otherwise be.

2.3 Data Preprocessing

Given a large, loosely organized collection of email, it needs to be transformed into a usable set of data. First, unique email messages are identified, and duplicate messages removed. Next, the email is grouped into threads. For simplicity, a group of email messages are considered a thread if they have identical subjects, except for prefixes such as RE and FWD, and no large gap in date between messages. Although this method might occasionally group together messages that are not actually part of the same thread, that a more detailed analysis of senders and recipients might avoid, this is done for safety and simplicity. Attempting to track senders and recipients to better identify separate email threads would cause a similar problem. Email messages that are actually part of the same thread would not be treated as such if the message chain went through people outside of the email domain of the dataset.

Along with the organization of email messages into threads, the detection of unique email addresses is done in a straightforward manner. The display name and the less than and greater than symbols are trimmed off email addresses in the header field.

No distinction is made between email addresses in the To, Cc, and Bcc fields. All email addresses in these fields are considered recipients of that email. If a recipient is included in multiple fields, or multiple times in the same field, they are not treated any differently than they would have been had their email address only occurred once.

2.4 Model Generation

After the dataset is preprocessed, a prediction model is generated that can be used to prioritize the email of users. Various approaches are taken to quantify the relationships, or more often lack thereof, between each pair of users and other persons. These methods are described in later sections. Once these relationships are quantified as a single numeric value, predictions can be made.

In a live system, email would be prioritized each time a user checked their inbox. Since these times are not available in any dataset that consists only of raw email messages, including the Enron email corpus, this information must be approximated based on messages that the user sent out. Specifically, it is assumed that if a user sends out an email message, they examined the contents of their inbox just prior to that. For a user's inbox at a particular point in time, values from the model are used to order messages from high priority to low priority. Since there is no user feedback in the dataset, the strength of a relationship between users must be determined from the email itself. To measure strength, the number of responses to email messages between users is examined. Specifically, shorter response times suggest a stronger relationship. That there are relationships where one user urgently reads all email messages sent by another, but rarely or never responds, cannot be satisfactorily taken into account. To do so would require users to explicitly provide this

via manual feedback, or an email browser that would track the order that messages were read in, and the amount of time spent reading each.

Chapter 3

Literature Review

3.1 Spam Detection

There are many different methods of doing simple spam detection. Some are entirely automated, while others require user input. One basic technique involves content-based filtering of the message body, and perhaps the title as well. Word or phrase based matching is done, possibly along with more complicated rule-based methods [1]. In both cases, the object is to filter out messages with words or phrases which indicate a high probability of being junk email. The downsides here are obvious. To start with, it is difficult to have a legitimate discussion about any topic which often appears in junk email. Furthermore, spammers will happily alter suspect words and phrases in order to get them past such a filter. These disadvantages are significant, and the literature suggests that probabilistic models are more effective [1].

Another often used approach to filtering spam is the application of blacklists. A blacklist is a collection of sites known to be sending out junk email. Email received from sites on the blacklist can be thrown out. While an effective tactic, this approach requires significant effort to maintain an adequate blacklist, or more likely, dependence on a third party to provide a correct, up to date blacklist [24]. Worse,

current research indicates that the effectiveness of blacklists is decreasing as spammers adopt more sophisticated techniques [24]. In particular, the use of botnets results in short lifespans for offending IP addresses [24]. It is suggested that in the long term, blacklists may do little beyond reducing the availability of proxy servers and open relays [24].

Email whitelists function in essentially the opposite manner of blacklists. A whitelist of approved email or IP addresses is maintained, and email messages from all senders that are not on the whitelist are assumed to be spam. Obviously, this method will eliminate all unsolicited email from appearing in the inbox. Unlike blacklists, the burden is placed on the user to maintain the whitelist, and this method virtually guarantees that valid email will be filtered into the spam folder [7]. Given the effectiveness of whitelists in eliminating all junk email, the lack of widespread adoption of this technique is a good indication that the cost of discarding all legitimate email from non-whitelisted email addresses is too much of a burden for most users. Like blacklists, whitelists may be combined with other methods [7].

Not exactly a cross between whitelists and blacklists as might be expected, greylisting is an authentication technique used to detect spam. Proposed by Harris, greylisting is an automatic method where the mail server stores a triplet containing the IP address, envelope sender address, and envelope recipient address for each incoming email message, rejecting the message if it has not recently seen that particular triplet [9]. The greylist itself is not merely a list of "good" or "bad" senders. It is assumed that a legitimate sender will attempt to resend the email message according to protocol upon seeing that the first attempt was rejected, at which point the message will be let through since its triplet is now on the greylist [9]. On the other hand, the assumption is made that a spammer will not attempt to resend an email

message that has been rejected [9]. Although this automated technique will never permanently reject a legitimate message, it does delay them, eliminating the near instantaneous nature of email [9]. Another obvious disadvantage is that greylisting could be circumvented completely if all spammers were to always retry delivery after a bounced message [9]. Although Harris does not fully address this seemingly fatal flaw, he notes that any movement towards software that retried delivery after receiving an error would at least increase the cost of spamming [9]. Furthermore, Harris suggests that a delay time of approximately one hour may be sufficient for blacklisting methods to flag the offending IP addresses [9].

Non-automated authentication filtering techniques also exist. Some challenge and response systems maintain a whitelist of permitted senders [22]. Rather than simply discard an email message from a sender not on the whitelist, a challenge response is sent to the sender of that message, requesting that some action be performed in the reply [22]. Once a a reply to the challenge is received, the original message is delivered [22]. Perone notes that these methods do not work with lists or automated email systems, and that any two people using such methods are in a state of deadlock, with no way to receive each others initial email message unless they first communicate via another method, and manually add each other to their whitelists [22]. Although a better protocol might resolve these issues, any CAPTCHA or other reverse Turing test required in the response could still be overcome via software methods. Barring a breakthrough in determining whether or not a subject is human, non-automated authentication solutions can at most increase the cost of sending spam.

A newer approach makes use of a Bayesian filter. Sahami et al. point out that the cost of incorrectly classifying a legitimate email as junk is significantly higher than misclassifying a junk email as legitimate [25]. Thus, it is important to try to avoid

throwing out legitimate messages that mention topics often occurring in junk email. Since a Bayesian classifier adapts based on user usage, it is an appropriate tool here. Research suggests that the use of Bayesian filters, when augmented with domain knowledge, is a highly effective tool.

Ultimately, a solution to spam may only come in the form of new email standards. Geer suggests replacing the SMTP standard with one that offers authentication, disallowing spammers the ability to mask their real identities [6]. A technological fix may not be forthcoming however. Spammers might set up their own domains and DNS servers, which would require additional functionality - whitelists, blacklists, etc. - to deal with [3].

3.2 Email Prioritization

Beyond spam detection lies email prioritization. Here, the focus is not on delineating messages into junk email and non-junk email, but rather ordering them according to how likely the user is to read and respond to each one. For ordering, each message must be assigned a numeric value. Probabilistic content-based filtering and Bayesian filtering already do this, although many of the other spam detection techniques are not suitable here.

Several researchers have investigated the use of social networks in spam detection and email prioritization. Boykin and Roychowdhury construct an algorithm based on the email history of a single user, where the resulting model is used to whitelist large connected graphs of friends [2]. The underlying assumption made is that those sending spam email will not know who the user's friends are, and so it is unlikely that a spam message will be sent with a friend as a co-recipient [2]. However, constructing

a model from the email history of a single user is inherently limited. Relationships between other pairs of users are determined only from email messages also sent to the user whose inbox is being used to construct the model, as messages from one third party to another are not available.

While Boykin and Roychowdhury are primarily concerned with spam detection, other researchers are using social networks for true email prioritization. One such recent method is the algorithm MailRank. Based on PageRank, it models social networks in an attempt to identify trusted senders, and so filter out spam [3, 21]. The authors provide two variants, Basic MailRank and Personalized MailRank. Both use the global email history to construct a model, however Personalized MailRank is more finely tuned in that the score of each email address is different for each MailRank user [3]. That is, the algorithm attempts to model the fact that a user might be much more or less important to another user than they are to the rest of the network. MailRank sets a threshold below which all incoming email is ignored as spam, and email above that threshold is prioritized according to its score [3].

Other algorithms involve user input for existing email messages in order to aid with prediction. Dabbish et al. note that an email may be important for the sender, but not the recipient, or vice versa, or important, but not urgent [5]. By increasing knowledge as to the nature of the relationship, actions may be predicted more finely. Email is not simply read, or discarded, but read and responded to, or read and responded to later, or read and not responded to later, or discarded, and so on. The paper stresses the importance of properly modeling the hierarchical structure of organizations. The authors conclude that the way an incoming message is handled is based not just on the raw importance of the message, but other factors as well,

such as the social status of the sender [5].

3.3 Enron Dataset

Although Enron filed for bankruptcy in 2001, the email dataset was not released until 2003 [8]. The original version of the dataset had various integrity problems that have since been fixed [14]. However, it is unclear exactly what these unspecified issues were. One noticable change is the fact that all email attachments in the dataset have been removed. As of 2008, the most recent version of the dataset is the March, 2, 2004 release, which is hosted by researchers at Carnegie Mellon University, and is available at http://www.cs.cmu.edu/~enron/. Regardless of flaws, due to the previous lack of a large, commonly available email dataset, much research has been done despite the relatively recent release, and generally slow time to publication [15]. However, not all of it is relevant to the topic at hand.

Klimt and Yang describe their preparation of the Enron email corpus in two papers [14, 15]. According to their work and other sources, after cleanup the corpus contained 200399 email messages belonging to 158 users, with a median number of 757 incoming and outgoing messages per user [12, 14]. These papers refer to the March 2, 2004 version of the dataset. However, it appears that version of the dataset has been further altered since then. As of late 2007, the March 2, 2004 version of the dataset contains exactly 150 users, so presumably further redaction has taken place.

Klimt and Yang do some statistical analysis in the first paper, noting that email messages are distributed roughly exponentially among users [14]. Furthermore, most users group their email into folders, and that the number of folders and email messages for each user have only a rough correlation with each other [14]. In the second

paper, they split the data in half chronologically for training and test sets, and proceed to use Support Vector Machines to classify email messages into folders, using the title and body of the message, as well as header information [15]. Their micro average F1 score, which is the harmonic mean of precision and recall, is around .7, while their macro average F1 score is only around .55, as it is dragged down by the users with many folders of low volume [15]. The authors also note the potential application of the dataset to spam filtering and email prioritization, although they focus on folder classification [15].

A highly relevant issue raised by Klimt and Yang is the difficulty of reassembling a collection of email messages back into threads. Messages with the same subject line sent between the same users are considered to belong to the same thread [15]. Messages with no subject line are not considered to belong to the same thread, regardless of the users involved [15]. Presumably, they also checked for subject prefixes such as RE and FWD, although they do not explicitly state this, nor whether email messages with the same subject but completely disparate dates were considered part of the same thread. They do state that they made no attempt to test the quality of their thread detection algorithm, and that the question of what constitutes a thread is subjective to begin with [15].

Lewis and Knowles tackle the issue of email threading by using the in-reply-to headers in the email messages [17]. Unfortunately, these are largely absent in the Enron email dataset [15]. Another method, by Murakoshi et al., treats thread detection as a natural language problem, using tree structures to represent email messages belonging to a particular thread [19]. This method is dfficult to use, and as expected, neither approach provides a particularly high degree of accuracy [17, 19]. Other techniques might be used, such as matching of pools of recipients and secondary

recipients in email messages suspected to be part of a thread, or checking the bodies of the messages for matching quoted text, but these methods would be tedious and similarly inexact. Hence, this thesis ends up using roughly the same methods as Klimt and Yang for thread detection.

Another version of the Enron dataset is available from the University of California at Berkely as a MySQL database. Hearst and his students in a natural language processing course annotated a subset of the Enron email corpus with category labels suitable for classification purposes [11]. Jabbari et al. also manually annotate a subset of the dataset into the categories Business and Personal [12]. Their interests are in automatic monitoring of personal email by businesses, a point of particular concern after court rulings stating that employee email may be monitored by an employer [12]. After removing junk email from the Enron dataset, they find that approximately 83% of the remaining messages are of a business nature, with the remaining 17% concerning personal matters [12].

Several researchers have made use of link mining techniques in order to detect community structure in the Enron email corpus. Using Singular Value Decomposition (SVD) and SemiDiscrete Decomposition (SDD) to evaluate the structure of the email, Keila and Skillicorn discovered a relationship with word usage and message length as well as relationships among individuals [13]. Specifically, they found that short messages tend to contain rarer words, while longer messages tend to contain more common words, a pattern they could not entirely account for [13]. Less surprisingly, their analysis showed that *individuals of similar status and role tend to communicate in similar ways* [13]. The authors also observed that changes in the corporate environment had a significant effect on word usage patterns [13]. More recent work by Murshed and Hossain examines the effect on subgroup structure during

organizational disintegration [20]. They find that subgroup cohesion may increase or decrease during a crisis, depending on whether a solution to the problem is available [20]. They conclude from the Enron dataset that the former happened, even though the organization ultimately broke apart entirely [20]. Although the period preceding the Enron collapse may be an extreme scenario, it is not unreasonable to suspect that this effect could cause difficulties with global email prioritization algorithms. Qian et al. use a link-based clustering algorithm to detect community structure in the Enron dataset [23]. Visual inspection of their results demonstrates that Enron executives tend to lie towards the center of the graph, with large clusters branching off from them [23].

Shetty and Adibi make use of the Korner definition of graph entropy in order to determine the organizational structure represented in the Enron email corpus [16, 26]. They are primarily interested in finding the most influential members in the network, and tracking the change in entropy of these group leaders over time [26]. Such statistics may be useful in the development of email prioritization algorithms, although we do not pursue such methods in this thesis.

Chapter 4

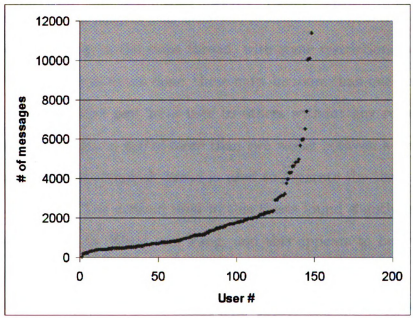
Local Email Prioritization

4.1 User Information

Enron was a large company, and the email dataset reflects that. As previously mentioned, the March 2, 2004 version of the Enron email corpus used in this thesis differs slightly from the March 2, 2004 version used in previous work, included those cited here. Comments included with the dataset indicate that some email messages have been redacted due to personal requests by those involved, and this may extend to entire inboxes [14]. There are 150 users, with an average of 23 folders each. Only 148 of these users actually have email messages, with a median of 1118 messages (Figure 4.1). The entire corpus contains 517431 total email messages. Once duplicate email messages are removed, 225484 unique messages remain. Interestingly, Klimt and Yang state that the original dataset was reduced from 619446 email messages to 200399 messages, after the removal of duplicates [14]. Their method for removing duplicates was to simply remove the folders *all_documents* and *discussion_threads* for each user [14]. What appears to be the case with the current dataset is that later redactions were made to a version of the the dataset prior to when these folders were removed, and then reposted without updating the version date. However, not all of the email messages in the two removed folders are duplicates. Thus, manually

removing duplicates results in slightly more unique messages, despite having 8 less users. Unlike previous classification efforts, the folders containing the email messages are not used in the generation of the prediction models here.

Figure 4.1: Email messages per user



A total of 28085 unique email addresses with the enron.com domain are seen in the dataset. Out of these, the email histories of 150 of them comprise the Enron dataset. It is not clear why specific employees were included or not, although they are generally higher up executives in the organization. Some key personnel are missing from this smaller group, such as Andrew Fastow, the chief financial officer of Enron Corporation. It appears likely that some of these people may have had their entire inboxes redacted. However, their email addresses do often appear in the email of others. Regardless of these issues, the corpus contains a significant chunk of real email from an organization.

4.2 Email Overview

Out of the 225484 email messages in the cleaned up dataset, 38860 total threads were detected. A total of 138568 email messages belonged to threads, for an average number of messages per thread of 3.6 (Table 4.1). Thread detection was done in a manner similar to Klimt and Yang [14]. A set of email messages where all messages have the same subject text after the prefixes RE and FWD are removed are considered to belong to the same thread, with some restrictions added. First, while complicated analysis is not done, there must be more than one author so that a series of email messages sent by a user to others without any responses is not considered a thread. Also, a gap of more than two weeks between a group of email messages constituting a potential thread is used to separate the the messages into two separate threads. The method used in this thesis found slightly more threads than the method used by Klimt and Yang, and this appears to be due primarily to the difference in handling of email with dates that are far apart. Some of the differences here are due to the usage of slightly different versions of the dataset as well.

Table 4.1: Distribution of email messages in threads

# messages	2	3	4	5	6	7	8	9	10-19	20+
# threads	21461	7700	3815	1910	1112	714	461	318	1024	346

Integrity issues were another concern. Although unspecified integrity problems were corrected to produce the March 2, 2004 version of the dataset, other concerns remain. The standard email address format at Enron was firstname.lastname@enron.com. For example, John Doe would be john.doe@enron.com. However, for many email addresses inside the enron.com domain, multiple versions of the otherwise identical email address are found in the corpus, with single quotation marks inserted at

arbitrary points in the local part of the email address. Although single quotation marks are valid characters in the local part of the email address, this behavior occurred with sufficient frequency that it appears the issue was caused by the Enron email system. Because of this, email addresses that are identical except for the occurrence of single quotation marks have been merged for testing purposes. Many email addresses appear in multiple forms due to ordinary typos, and these were left unchanged.

Dates on email messages show some interesting and irregular behavior as well. The email corpus covers a period from approximately January 1, 1997 to July 12, 2002. Most of the email messages are from the beginning of 2000 on, however. It is not clear whether email use was simply not as widespread at Enron during the 1997-1999 period, or whether email for most of the users during this time was not included in the dataset. Also, a small percentage of legitimate as well as spam messages have obviously invalid dates. Most of these are either January 1, 1980, or dates that were in the future at the time that the dataset was released. These were simply left as is for testing purposes.

4.3 Model Creation

Once duplicate email messages are removed and the data is otherwise preprocessed, the relationships, or links, between users and other persons in the database need to be constructed. In the local model, these links are created entirely from email messages sent between a pair of people. A personalized prioritization system could then be built from the email history of a single user.

It is trivial to count the number of email messages sent from one person to another. However, simply sending a lot of messages to a person, especially if relatively few messages were received from that person, is not necessarily indicative of a strong relationship. Instead, replies are tracked, on the assumption that a high reply rate and low response times are more suggestive of a significant relationship between two people. Furthermore, considering the replies to messages that have been sent allows for testing the performance of prioritization algorithms used.

Given that thread detection is imperfect, it comes as no surprise that determining the hierarchy of email messages within a thread is also inexact. If person A sends an email message to person B, and person B later composes a new message from scratch with the same subject text, there is no way to determine that the message B sent was not in fact a response. Although this is a minor issue, other more significant situations arise. For example, if person A sends two email messages in a thread to person B, and person B sends a reply, it is unclear which message is being replied to. In some cases, it could be both. For simplicity, it is assumed that any reply sent from person B to person A is a response to the latest email that person A sent to person B. Adding to this complexity is the fact that an email can have multiple recipients. Remember that recipients, secondary recipients, and blind secondary recipients are all considered recipients in this context. So if person A sends an email message to persons B and C, after which person B forwards the message to person C, then any message sent by person C to persons A and B will count as a reply to both.

Deciding which email message a reply is a response to is necessary in order to calculate the response time of a reply. While a high response rate to another user is more suggestive of a strong relationship than simply receiving a lot of unanswered email messages, having fast response times are more significant still. In prioritizing email,

the goal is not simply to detect spam, nor even to determine which messages in the inbox are most important, but which messages in the inbox are the most urgent at a given point in time.

Once the threads in the email corpus are reconstructed, and responses in them are identified, a graph can be created for the purpose of email prioritization. Each person seen in the Enron dataset is a node in the graph. This includes the 150 users whose email history comprises the dataset, along with the other 27935 persons in the enron.com email domain who appear in these email histories, as well as the 49533 persons outside the enron.com email domain that are also present.

Links between nodes are constructed in a straightforward manner. They contain a considerable amount of information, although ultimately this must be condensed into a single value that can be used for ranking. The graph is a directed graph, so each link is in fact an arc, and each pair of nodes has an arc going each way. While this is a lot of links, the graph is actually fairly sparse, as the email corpus only contains 338441 relationships between person A and person B such that person A has received email from person B. As a result, even in the global email prioritization models seen in the next chapter, most of the links contain no information, and have nothing to compute.

The links that do represent an actual relationship between two people are created by taking the number and identity of email messages the tail node has sent to the head node, the head being the node that the link is pointed towards. From here, the number of responses to these replies sent by the head back to the tail node is computed. This process is done according to the definition of a response that was discussed previously. Note that the number of responses for a given pair cannot

exceed the number of messages sent out from the tail node to the head node. For local email prioritization, only the email history of a particular user is made use of in creating a model.

Along with the number of replies, the total response time of all of the replies is computed. The response time of a single response is the timestamp of the original email subtracted from the timestamp of the response email. Similarly, the total response time is simply the sum of the response times of each response seen in the link. The average response time of the replies in the link is calculated from this by dividing the total response time by the number of responses seen. This average response time is then smoothed, using standard smoothing methods, so that the values for links with few response times is not extremely misrepresented on the basis of one or two very fast or very slow responses. As email messages with very large date gaps are not considered to belong to the same thread, these values should not be too extreme towards the direction of unrealistically long response times to begin with.

The smoothed average response times are the values actually used for the prioritization of email. In the models used for local email prioritization, only the email history of a single user is known, so the only links that are easily used in prediction are those incident on the node representing that particular user. As an email can be sent to multiple recipients, the email history of a single user will contain some portion of the email messages between other people. However, an email message and corresponding response between two other people will only be detected if both messages are also sent to the user whose email history is being used to build the local model. As such, the information in these links is extremely incomplete, and is not used here to assist with local prediction.

The prediction models generated for both local and global email prioritization algorithms are simple digraphs. Of course, having multiple links from one node to another does not make sense when a link represents the email relationship between the two people it is incident on. However, just as a person might send or forward an email message to a secondary email address of theirs, they might also send it to the same email address the message is being sent from. This may be done for several reasons. The person's email address might be included in a list an email is being sent to, or they might explicitly include themselves as a recipient so that they can reference the email message later, or they may be using it as a method to transfer files, by including them as attachments to a message sent to themselves. In all cases, it is unlikely that a person will reply to an email sent to themselves. So although loops in a prediction model could be handled in the same manner as normal links, it does not make sense to include this information and use it for email prioritization. In a real system, such messages might be handled by arranging them at the top of the inbox, followed by the rest of the email messages, ranked as normal.

4.4 Model Examination

Examination of the prediction models generated for local email prioritization shows some interesting statistics. Not surprisingly, the Enron executives whose email histories comprise the database tend to have significant email relationships with each other. Jeff Dasovich, Enron's government affairs executive, and the individual with the highest amount of email messages in the corpus, is one of the nodes in several of the links with the highest number of email messages sent. The strongest link, between Dasovich and Susan Mara, another Enron executive, saw 723 email messages sent over the period of roughly two years. Most of the links between the group of 150 users saw much less traffic, with the number of messages sent in the double

digits to low triple digits. Unfortunately, for many of these high volume links, most of the messages received were sent out to a wide group of Enron employees, and there were few responses. The link between Tana Jones to Leslie Hanson contained the highest number of responses, with 178 responses to 337 email messages sent. Interestingly, Leslie Hanson is not in the group of 150 users. Of course, this number was exceptionally high. Even most links that did have a considerable amount of mail sent still had a single digit or low double digit number of actual responses.

In order to evaluate the models, the email corpus is separated into a training set and a test set. The first 70% of the email messages, according to date, are used as training data, and the remaining 30% of the messages are used as test data. Local prediction models are then built from the training data. However, this means the link information used to create the models is somewhat weaker than discussed in the previous paragraph.

The test data is divided into incoming and outgoing messages for each user. The outgoing messages are further divided into clusters based on their timestamps. Each of these outgoing clusters represents a point when the user checked their inbox. The group of incoming messages between the current outgoing cluster being examined and the previous one is considered to be the inbox at that point, and predictions are made for each such group.

For a cluster of outgoing email messages and corresponding inbox, messages in the inbox which did not receive a response are ignored. Then, predictions are made for the email messages in the inbox. For the local prioritization done here, each email message received by the user is assigned the smoothed average response time of the author of that email. The messages in the inbox are then ranked according

to increasing values, with the ties broken by email timestamps. That is, if the value assigned to two incoming email messages is the same, the message with the earlier timestamp is ranked higher than the message with the later timestamp. Once this is done, the predicted ranking of the messages for which a reply is sent out is compared to the actual order of the outgoing messages in that cluster.

4.5 Prediction Results

The accuracy of the predictions for an outgoing cluster of email messages is calculated by comparing the order of each pair of outgoing messages to their order in the predicted rankings. If the order is the same, the pair is considered to have been correctly classified. If the pair is out of order in the predicted rankings, the pair is considered to have been incorrectly classified. The results of the predictions for each outgoing cluster of email for a user are then summed together.

Clusters with only a single outgoing email message are ignored when doing classification, as there are no predictions to be made. An outgoing cluster with two outgoing email messages has only one pair that needs compared, since the order of the first email message and the second email message will be in order in the predicted rankings if and only if the order of the second email message and the first email message are in order in cluster of outgoing messages. Similarly, there is no point in comparing the order of the first email to itself, as they will occupy the same position, and by definition, cannot be out of order. Note that prediction using randomized rankings according to these procedures would result in approximately 50% of the instances being correctly classified.

Since predictions are only made for the last 30% of the email messages in the dataset, and responses are relatively low compared to the total number of email messages received, relatively few classifications are made, at least compared to the original size of the email corpus. This is exacerbated by the fact that many clusters of outgoing email messages only contain a single message, and so no predictions are made for those clusters. Of the 150 users that local prediction models are built and tested for, many of them have a small number of predictions made. Most users have only a few predictions, while a large minority of users that actively used email account for the majority of predictions made. The user with the most predictions made was Jim Steffes, with 432 total pairs classified. A few users with very light email usage had none at all.

When the predictions of all users are summed together, the resulting accuracy is somewhat discouraging. A total of 2939 pairs were classified. In 1526 cases, or 51.9%, they were classified correctly, while the remaining 1413, or 48.1%, were classified incorrectly. While this is somewhat better than 50%, it is not clear that the predictions are statistically better than the results of random predictions.

Chapter 5

Global Email Prioritization

5.1 Overview

When doing local email prioritization, the model is limited solely to the email history of a single user. The primary benefit of using local models is that they are much easier to create and maintain. Building any global model requires the email history of other users, and even if they are all other people in the same email domain, this is more complicated technically, and can raise privacy issues as well. However, from the perspective of building good models, it is always beneficial to have more information available than less. When restricted to the email history of a single user, only a highly incomplete picture of the relationships between pairs of people who both have a relationship with the user is seen. Worse, these relationships will only be detected so far as email messages between other people include the user for whom the model is being built for as a recipient. Email messages that simply go back and forth between other people the user has a relationship with are not seen. Furthermore, no information at all can be known about any person with no relationship to the user, since they do not appear in the dataset used to build a local model. So, attention is instead focused on constructing a graph that is usable for the purpose

of global email prioritization.

5.2 Model Creation

Creating a model to use for global email prioritization was done in a fairly similar manner as creating models for local email prioritization. Preprocessing is done, after which thread detection and identification of response messages is done in exactly the same manner as before. The model is again a simple digraph, and nodes and links are computed, with same information included in the links. Once the average response times for each link have been computed, and smoothed, creation of the global model diverges from that of the local model.

Instead of ranking each email message from a sender according to the raw smoothed average response time between the user and that sender, the goal is to factor in the extent to which other users view the senders messages as important or urgent. To do this, the smoothed average response times of all other users and the sender is averaged. These averaged values are computed for every user. Once this is done, these average smoothed average response times are factored into the raw smoothed average response times for each link, with the new averages weighted equally against the raw values, or otherwise as desired. Thus, the final values used as scores to order email messages are based on significantly more data than in the local model.

5.3 Model Examination

After the refactored smoothed average response times are computed, the portion of the dataset set aside for testing is handled in the same manner as it was using

local email prioritization. A user's outgoing email messages are clustered according to their timestamps, and incoming messages are sorted into corresponding inboxes. Each cluster and related inbox is then examined one at a time, and the messages in the inbox are ranked according to the new smoothed average response times. As before, any ties in ranking between two email messages are broken using their timestamps. After ranking is complete, the accuracy of the predictions can be tested.

The model created for global email prioritization is not that different from the models generated for local email prioritization. The difference is that when the email history of all users is available to build a model, the relationships of others can be reasonably taken into account. Since the smoothed average response times are simply averaged to create composite values used for ranking, the new values tend to look fairly similar, as they are simply based on more data.

5.4 Prediction Results

A global prediction model was generated for the Enron dataset, with the average response times computed by weighting the personal value and the global average equally. Classification was again done by comparing the order of each pair of email messages in a ranked inbox to their order in the corresponding cluster of outgoing messages, and tallying the number of correctly and incorrectly classified pairs. However, when the predictions of all users were summed together, the percentage of pairs classified correctly was not significantly higher than it was using local email prioritization. Out of 2939 total pairs classified, 1536, or 52.3% were classified correctly, while 1403, or 47.7% were classified incorrectly.

While the results here are a slight improvement over those generated using the local email prioritization models, they are still not significantly better than the 50% that would be expected simply by prioritizing email messages in the inbox in random order. It is not entirely clear why this is the case, as the global prediction model uses much more data than the local prediction model.

Several explanations might account for the poor performance of the global email prioritization algorithm used here. The most obvious is that the fault may lie with the algorithm itself. While it is difficult to believe that responses are not indicative of a high priority email message, faster response times may not show a correlation with message performance. Non-spam, but relatively lower priority email messages may tend to receive quick, but short responses, while higher priority messages may tend to receive more lengthy responses that take longer to compose, or are written at a later date. Another possibility is that the small number of time slices with multiple responses that could be used for prediction was not an adequate number to provide meaningful results.

Chapter 6

Conclusion

6.1 Analysis

A number of issues play havoc with any attempt to create an email prioritization system. Thread detection was the most problematic, and this thesis made no significant improvements over the methods Klimt and Yang used with their work on the Enron dataset [15]. As threads are not able to be perfectly reconstructed, there appears to be no way to eliminate this issue as long as thread information is used in building a model. Previous research partially avoided the problem by focusing on the classification of email messages in the Enron dataset according to their folders. However, this is not really email prioritization - it is much closer to spam detection with a non-boolean class attribute. Although it can be assumed that both the folders *business* and *golf* contain email messages that are of interest to the user, focusing on folders does not really allow for properly judging the relative importance of various types of messages.

The difficulty involved in determining background information such as when users checked their inboxes was another major hurdle. Unfortunately, there is no large, publicly available dataset available at present that contains such augmenting in-

formation along with the email corpus itself. While this thesis pursued entirely automatic email detection, it appears that requiring manual user feedback might be required to solve some of these issues. While this might not be problematic for users simply seeking good spam detection, requiring more work from users interested in email prioritization in order to relieve an overwhelming amount of incoming email messages is a difficult proposition.

Another troublesome issue was how to properly deal with the incomplete information in links. For local email prioritization, only the links incident to the user whose email history is being used to build the model is complete, while links between others may understate a relationship between two people, or indicate that none at all exists, when that is not the case. For global email prioritization, links between all of the users have complete information, while links between other people do not. In both cases, it is unclear how to factor the information in these relationships into a model, as responses between pairs people that are not users are only rarely seen.

6.2 Future Work

Despite the size of the Enron email corpus, many of the email messages do not belong to threads, and fewer yet are responses to messages. The result of this is that the amount of training and testing data available to build models is actually quite small. Furthermore, it is unreasonable to require years of email data before accurate predictions can be made. So only using data from email responses appears to be inadequate. The easiest way to avoid this is to focus on classifying messages by folders, as much of the previous work has done. However, this approach is somewhat limited, as each user has different folders, and true email prioritization is not being done here.

There are several alternatives to the use of folders. One of these is adding user feedback to the email client and using that data in the building of models. This has the clear disadvantage of restricting users to a particular email client. More significantly, it is felt that making email use more complicated for users is ultimately a dead end, especially when the goal is to reduce the time spent on email. Another is to use more data from the email message itself. Clearly, raw response data is inadequate, although creating an accurate email prioritization algorithm solely from header information may still be possible. More likely, text from the body of the email must be used as well.

APPENDIX

APPENDIX

Table 1: User statistics

Email address	Messages sent	Messages recvd.	Responses sent
phillip.allen@enron.com	386	795	1
john.arnold@enron.com	910	965	24
harry.arora@enron.com	76	465	5
robert.badeer@enron.com	83	1068	3
susan.bailey@enron.com	230	2031	24
eric.bass@enron.com	1346	913	13
don.baughman@enron.com	192	1180	7
sally.beck@enron.com	1528	3303	36
robert.benson@enron.com	21	460	1
lynn.blair@enron.com	970	1603	47
sandra.brawner@enron.com	111	292	1
rick.buy@enron.com	560	1577	47
larry.campbell@enron.com	381	740	15
mike.carson@enron.com	241	573	5
michelle.cash@enron.com	1169	1170	47
monika.causholli@enron.com	463	1717	13
shelley.corman@enron.com	691	1478	32
sean.crandall@enron.com	158	869	12
martin.cuilla@enron.com	135	312	5

Email address	Messages sent	Messages recvd.	Responses sent
jeff.dasovich@enron.com	4674	6854	633
dana.davis@enron.com	297	1092	10
clint.dean@enron.com	41	538	2
david.delainey@enron.com	722	873	3
james.derrick@enron.com	715	961	32
stacy.dickson@enron.com	214	904	10
tom.donohoe@enron.com	34	280	1
lindy.donoho@enron.com	271	1246	7
chris.dorland@enron.com	661	425	11
frank.ermis@enron.com	28	438	0
daren.farmer@enron.com	734	2299	18
mary.fischer@enron.com	81	115	10
m..forney@enron.com	387	328	12
drew.fossum@enron.com	1109	866	25
lisa.gang@enron.com	92	619	17
randall.gay@enron.com	173	332	1
tracy.geaccone@enron.com	567	935	34
chris.germany@enron.com	3275	1643	22
doug.gilbert-smith@enron.com	111	787	3
darron.giron@enron.com	747	238	1
john.griffith@enron.com	124	781	13
mike.grigsby@enron.com	633	1114	20
mark.guzman@enron.com	293	4665	0
e..haedicke@enron.com	129	504	14
mary.hain@enron.com	474	1363	14
steven.harris@enron.com	106	1893	19
rod.hayslett@enron.com	671	1311	74
marie.heard@enron.com	841	1325	46

Email address	Messages sent	Messages recvd.	Responses sent
scott.hendrickson@enron.com	67	474	11
judy.hernandez@enron.com	216	324	0
john.hodge@enron.com	44	568	9
keith.holst@enron.com	38	634	2
stanley.horton@enron.com	437	1032	16
kevin.hyatt@enron.com	584	1452	25
dan.hyvl@enron.com	681	1076	71
tana.jones@enron.com	4092	5959	567
vince.kaminski@enron.com	3659	3680	8
steven.kean@enron.com	1266	3354	18
peter.keavey@enron.com	61	174	1
kam.keiser@enron.com	414	1232	35
jeff.king@enron.com	20	511	3
louise.kitchen@enron.com	1162	2008	46
tori.kuykendall@enron.com	250	402	10
lavorato@enron.com	267	116	2
kenneth.lay@enron.com	20	1673	2
matthew.lenhart@enron.com	1463	957	36
andrew.lewis@enron.com	27	265	0
eric.linder@enron.com	11	1566	2
lokay@bigfoot.com	25	13	0
teb.lokey@enron.com	136	658	24
phillip.love@enron.com	837	423	15
t..lucci@enron.com	208	212	5
mike.maggi@enron.com	39	381	3
kay.mann@enron.com	4598	2886	369
thomas.martin@enron.com	30	402	0

Email address	Messages sent	Messages recvd.	Responses sent
larry.may@enron.com	71	387	9
danny.mccarty@enron.com	162	593	10
mike.mcconnell@enron.com	730	1069	4
brad.mckay@enron.com	95	480	3
jonathan.mckay@enron.com	235	719	34
errol.mclaughlin@enron.com	454	1084	25
steven.merris@enron.com	4	435	1
albert.meyers@enron.com	44	2324	0
patrice.mims@enron.com	222	244	1
matt.motley@enron.com	14	738	0
scott.neal@enron.com	587	1320	23
gerald.nemec@enron.com	2123	3591	232
stephanie.panus@enron.com	398	1536	36
joe.parks@enron.com	449	943	55
susan.pereira@enron.com	125	193	1
debra.perlingiere@enron.com	2002	947	45
vladi.pimenov@enron.com	63	354	11
phillip.platter@enron.com	116	680	3
m..presto@enron.com	734	1083	60
joe.quenet@enron.com	81	282	2
dutch.quigley@enron.com	392	780	30
bill.rapp@enron.com	122	348	12
jay.reitmeyer@enron.com	61	424	11
cooper.richey@enron.com	272	230	6
andrea.ring@enron.com	133	405	5
richard.ring@enron.com	85	610	10
robin.rodrigue@enron.com	599	246	7

Email address	Messages sent	Messages recvd.	Responses sent
benjamin.rogers@enron.com	447	1511	14
kevin.ruscitti@enron.com	215	428	14
elizabeth.sager@enron.com	1455	2345	62
eric.saibi@enron.com	30	560	2
holden.salisbury@enron.com	150	1126	26
monique.sanchez@enron.com	111	294	5
richard.sanders@enron.com	1525	1652	29
diana.scholtes@enron.com	161	906	6
darrell.schoolcraft@enron.com	474	886	15
jim.schwieger@enron.com	169	570	11
susan.scott@enron.com	1113	1209	31
cara.semperger@enron.com	474	687	23
sara.shackleton@enron.com	4403	5685	341
a..shankman@enron.com	200	808	16
richard.shapiro@enron.com	436	5604	58
s..shively@enron.com	137	635	13
jeff.skilling@enron.com	58	1474	11
ryan.slinger@enron.com	71	4233	1
matt.smith@enron.com	435	532	23
geir.solberg@enron.com	78	4256	0
steven.south@enron.com	20	159	0
theresa.staab@enron.com	147	337	17
carol.clair@enron.com	1442	1498	58
d..steffes@enron.com	1418	1806	113
joe.stepenovitch@enron.com	61	805	6
chris.stokley@enron.com	6	232	0
geoff.storey@enron.com	125	648	16

Email address	Messages sent	Messages recvd.	Responses sent
fletcher.sturm@enron.com	105	331	0
mike.swerzbin@enron.com	65	787	6
kate.symes@enron.com	1475	2489	132
mark.taylor@enron.com	1912	4668	156
jane.tholt@enron.com	230	239	3
d..thomas@enron.com	137	639	19
judy.townsend@enron.com	58	636	4
barry.tycholiz@enron.com	554	1168	36
kim.ward@enron.com	253	469	29
kimberly.watson@enron.com	995	2086	100
v.weldon@enron.com	230	163	9
greg.whalley@enron.com	159	1859	12
w..white@enron.com	470	1395	54
mark.whitt@enron.com	420	711	40
jason.williams@enron.com	123	455	3
bill.williams@enron.com	649	4237	55
jason.wolfe@enron.com	90	746	19
paul.ybarbo@enron.com	189	974	21
andy.zipper@enron.com	381	1336	65
john.zufferli@enron.com	306	293	15

REFERENCES

REFERENCES

[1] N. J. Belkin and W. B. Croft. Information filtering and Information Retrieval: Two sides of the same Coin? *Communications of the ACM*, 35(12):29-38, December 1992.

[2] P. O. Boykin and V. Roychowdhury. Personal Email Networks: An Effective Anti-Spam Tool. Preprint, February 2004. http://www.arxiv.org/abs/cond-mat/0402143.

[3] P. A. Chirita, J. Diederich, and W. Nejdl. MailRank: Using Ranking for Spam Detection. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, 373-380, October - November, 2005.

[4] J. C. Coffee. What caused Enron? A capsule social and economic history of the 1990's. Working Paper 214, 2003.

[5] L. A. Dabbish, R. E. Kraut, S. Fussell, and S. Kiesler. Understanding Email Use: Predicting Action on a Message. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, 691-700, April 2005.

[6] D. Geer. Will New Standards Help Curb Spam? *Computer*, 37(2):14-16, February 2004.

[7] J. Golbeck and J. Hendler. Reputation Network Analysis for Email Filtering. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004.

[8] T. Grieve. The decline and fall of the Enron empire. *Salon*, October 2003. http://dir.salon.com/story/news/feature/2003/10/14/enron/index.html.

[9] E. Harris. The Next Step in the Spam Control War: Greylisting. White Paper, August 2003. http://projects.puremagic.com/greylisting/whitepaper.html.

[10] P. M. Healy and K. G. Palepu. The Fall of Enron. *Journal of Economic Perspectives*, 17(2):3-26, Spring 2003.

[11] M. Hearst. UC Berkeley Enron Email Analysis. Website. http://bailando.sims.berkeley.edu/enron_email.html.

[12] S. Jabbari, B. Allison, D. Guthrie, and L. Guthrie. Towards the Orwellian Nightmare: Separation of Business and Personal Emails. In *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, 407-411, 2006.

[13] P. S. Keila and D. B. Skillicorn. Structure in the Enron Email Dataset. *Workshop on on Link Analysis, Counterterrorism and Security, SIAM International Conference on Data Mining*, 55-64, April 2005.

[14] B. Klimt and Y. Yang. Introducing the Enron Corpus. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, July 2004.

[15] B. Klimt and Y. Yang. The Enron Corpus: A New Dataset for Email Classification Research. In *Proceedings of ECML'04, 15th European Conference on Machine Learning*, 217-226, 2004.

[16] J. Korner. Bounds and Information Theory. *SIAM Journal on Algorithms and Discrete Mathematics*, 7(4):560-570, 1986.

[17] D. D. Lewis and K. A. Knowles. Threading Electronic Mail: A Preliminary Study. *Information Processing and Management*, 33(2):209-217, 1997.

[18] B. McLean and P. Elkind. *The Smartest Guys in the Room.* The Penguin Group, New York, New York, 2003.

[19] H. Murakoshi, A. Shimazu, and K. Ochimizu. Construction of Deliberation Structure in EMail Communication. *Computational Intelligence*, 16(4):570-577, 2000.

[20] S. H. Murshed and L. Hossain. Exploring Interaction Patterns of Cohesive subgroups during Organizational Disintegration. In *Proceedings of the 7th ACM SIGCHI*, 254:59-66, 2007.

[21] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report, Stanford University, 1998.

[22] M. Perone. *An Overview of Spam Blocking Techniques.* Technical Report, Barracuda Networks, 2004.

[23] R. Qian, W. Zhang, and B. Yang. Detect Community structure from the Enron Email Corpus Based on Link Mining. In *Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications*, 2:850-855, 2006.

[24] A. Ramachandran, N. Feamster. Understanding the Network-Level Behavior of Spammers. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 291-302, September, 2006.

[25] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian approach to filtering junk email. *AAAI Workshop on Learning for Text Categorization*, 55-62, July 1998.

[26] J. Shetty and J. Adibi. Discovering Important Nodes through Graph Entropy: The Case of Enron Email Database. In *Proceedings of the 3rd International Workshop on Link Discovery*, 74-81, 2005.

[27] R. R. Sims and J. Brinkmann. Enron Ethics (Or: Culture Matters More than Codes). *Journal of Business Ethics*, 45(3):243-256, 2003.