Intelligent Email: Reply and Attachment Prediction

Mark Dredze, Tova Brooks, Josh Carroll
Joshua Magarick, John Blitzer, Fernando Pereira

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA
mdredze,tmbrooks,carrollk,magarick,blitzer,pereira@seas.upenn.edu

ABSTRACT

We present two prediction problems under the rubric of Intelligent Email that are designed to support enhanced email interfaces that relieve the stress of email overload. Reply prediction alerts users when an email requires a response and facilitates email response management. Attachment prediction alerts users when they are about to send an email missing an attachment or triggers a document recommendation system, which can catch missing attachment emails before they are sent. Both problems use the same underlying email classification system and task specific features. Each task is evaluated for both single-user and cross-user settings.

ACM Classification Keywords
H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General Terms
Design, Human Factors.

Author Keywords
email overload, reply prediction, attachment prediction

INTRODUCTION

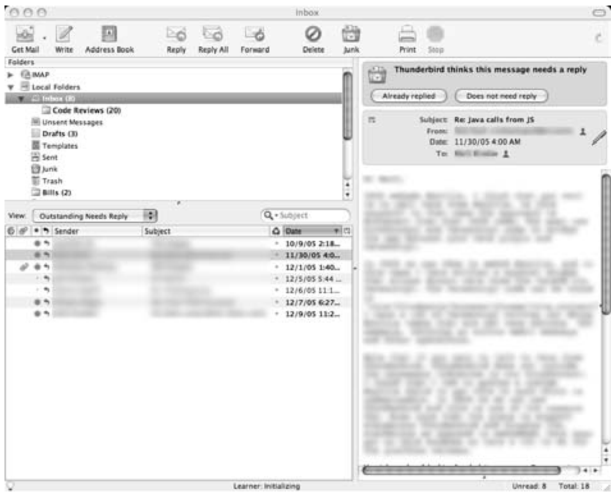
Numerous articles have appeared in the media discussing the problem of email overload. The convenience of email to reach anyone at any time coupled with its central role in any modern organization has led to an explosion in the sheer volume of messages a person receives. A person must handle a large amount of email as part of her job and it has evolved to encompass a plethora of work-related activities. Whittaker and Sidner [8] analyzed the use of email to perform task management, personal archiving, and asynchronous communication, and referred to the three as "email overload." They concluded that users perform a large variety of work-related tasks with email, becoming overwhelmed by the amount of information in their mailboxes. In the past decade, HCI research has studied the problem of email overload and explored new interfaces [7]. Similar efforts have emerged in the AI and machine learning communities, including social information to improve email triage [4], recommending message recipients [1], classifying email according to its intention [2] and automatically sorting email into folders [5]. These applications have been termed Intelligent Email.

This work explores two new Intelligent Email prediction tasks that improve the email user experience: *reply prediction* and *attachment prediction*. Both use machine learning to assist overwhelmed email users and support an enhanced email interface. We discuss the general prediction system, then explore the goals, features and evaluations for each task.

SYSTEM SETUP

Reply prediction and attachment prediction, as well as other email prediction tasks, can be treated as binary classification problems, where for each email the system outputs a label, either positive (*needs reply* or *needs attachment*) or negative (*does not need reply* or *does not need attachment*). Each email is represented as a sparse, high-dimensional binary vector $\mathbf{x} \in \{0,1\}^D$, where each dimension represents a feature. Real valued features are quantized for representation in this format. Learning is performed with a logistic regression classifier, which achieved better performance than other trainable classifiers on preliminary experiments. Email classification, studied for a number of other tasks, typically relies on a bag of words representation of the subject and body with history removed, as well as the recipients and sender of the message as features [1, 2, 5]. These baseline features are included in our system.

Both systems are evaluated using F_1, the harmonic mean of precision and recall, standard metrics for classification tasks. In each evaluation, both single-user and cross-user (adaptation) settings are considered. Typically, learning systems are trained on user-specific data, learning features particular for each user. This is the standard single-user setting. However, when predictions are required for a new email user, there are no existing data or use-provided labels for training. The cross-user setting evaluates this scenario, determining the performance of a system "out of the box."



Subject: Re: Java calls from JS
Date: 11/30/05 4:00 AM

10/9/05 2:18...
11/30/05 4:0...
12/1/05 1:40...
12/5/05 5:44 ...
12/6/05 11:1...
12/7/05 6:27...
12/9/05 11:2...

Figure 1. The reply management interface in Thunderbird has an additional column (4th from left) in the bottom left pane that displays a reply arrow next to messages that need a reply. Two buttons above the message contents read: "Already replied," which marks the response as fulfilled and "Does not need a reply." Messages that still need a reply are viewed by selecting the "Outstanding Needs Reply" view (shown).

REPLY PREDICTION

A quotation from interviews conducted by Whittaker and Sidner [8] captures a particularly familiar complaint regarding email as a communication medium.

"Waiting to hear back from another ... employee can mean delays in accomplishing a particular task, which can ... have significant impact on our overall operations. ... it can be critical or just frustrating."

"One of my pet-peeves is when someone does not get back to me, but I am one of the worst offenders. I get so many emails ... that I cannot keep up."

Reply management addresses this problem by providing the user tools to manage outstanding reply commitments. Our prototype interface both provides information by indicating which messages require a reply, and allows the user to manage these messages, marking them as replied or needs reply and displaying all outstanding reply commitments. A screen shot of the interface as implemented in the Mozilla foundation's Thunderbird mail client is shown in figure 1. The reply predictor, the intelligent interface component, automatically identifies and marks new messages that need a reply.

Features

Consider a user who sends a message to Merrick and Tahlia with the subject "Visiting client schedule." Who should reply? If Tahlia is only CCed on the message, then it is likely a reply is not expected. However, Merrick, who was the primary recipient, needs to reply. The exact same message receives two different classifications, an impossible task using message content features alone. Instead, the best indicator of an action is the user's relationship with the message and sender. The reply predictor uses *relational features* that rely on a user profile, constructed from the training data for each user. A user profile includes the total number of sent and received messages to each user, the user's address book, supervisor (user-provided), and email address and domain. Some features constructed using this profile include *I appear in the CC list*, *I frequently reply to this user*, and *sender is in address book*. Relational features have two advantages. First, general behavior patterns can be learned, such as establishing tendencies towards address book contacts in general, which extends to newly added contacts. Second, a system trained on a user can be adapted to a new user without training data (user adaptation) since specific email addresses and names are not used as features.

Additional document-specific features were used. These features include the presence of question marks in an email. Other features are used to capture requests without question marks, such as, "Can you please send me a report today." The system examines all words in the training data and finds those that best indicate a question through their TF-IDF scores, where the term frequency is the number of times the word appears in a question and the document frequency is the total number of sentences containing the word. The 30 words with the highest TF-IDF scores are termed *question words* and the feature *has question word* is added where appropriate. Additional features included the presence of an attachment, document length, salutations, and time of day.[1]

For the email evaluated in this study, these feature rules produced roughly 15,000 binary features, although the vast majority were from the bag of words features; less than 200 were relational or task specific.

Corpus

The reply predictor was evaluated on the mailboxes of four users, Merrick, Jarvis, Jaeger and Tahlia.[2] Each user hand labeled his or her messages as either *needs reply* (positive) or *does not need reply* (negative). Manual labels were used instead of automatically extracted labels from previous behavior since this behavior was found to be erratic and inconsistent with what users identified as the correct action. It is not surprising that overwhelmed users acknowledge that a message did require their reply even though they failed to do so; classifiers trained on actual user reply behavior are thus very poor. The datasets represent received messages that went to the user's inbox and not those automatically filtered into other folders upon arrival, excluding most automated messages, listserv mail and spam. The corpus included 2391 total emails, where each user's mail ranged over 1 to 6 months and included between 443 and 741 emails, with about one-third needing a reply.

Evaluation

A rule based baseline system was used to predict reply labels based on the presence of question marks, the words "reply " or "urgent" and if the sender is in the address book. These

[1]It is impractical to list every feature rule due to space constraints, but a full list and implementation is available upon request.

[2]Public corpora, such as the Enron corpus, lack full headers and reply data needed for this task.

User	*Recall*	*Precision*	F_1
Jaeger	0.42	0.55	0.47
Jarvis	0.67	0.77	0.71
Merrick	0.68	0.83	0.74
Tahlia	0.77	0.76	0.77
Average	0.64	0.73	0.67

Table 1. Results for 10 randomized trials for the reply prediction system on four email users using all available features.

Test	*Self*	*Recall*	*Precision*	F_1	Δ
Jaeger	0.47	0.56	0.72	0.63	+0.16
Jarvis	0.71	0.52	0.86	0.65	-0.06
Merrick	0.74	0.68	0.73	0.71	-0.03
Tahlia	0.77	0.77	0.59	0.67	-0.10

Table 2. A cross-user evaluation for four users averaged over 10 randomized runs. Each classifier is trained on three users and tested on the fourth using relational and task specific features. Δ is the F_1 difference between this test and *Self*, the full classifier trained on just the user's data.

rules yielded an F_1 of 0.52 averaged across 10 runs, clearly indicating that simple rules are insufficient. The learning system and additional features described above were evaluated on each of the four users across ten randomized runs with splits of 80/20 train/test. Results for each user are shown in table 1. Precision for this task is much higher than recall, indicating that there are many messages that are difficult to identify as needing a reply. To gauge the difficulty of the task, inter-annotator agreement was measured as the F_1 on Tahlia and Jaeger's email. The resulting score of 0.77 is close to the results for most of the users. The reason for the poor performance of Jaeger's mail was not clear despite some investigation.

The reply predictor was also evaluated in the cross-user setting by holding out a single user and training on the remaining three. The baseline features – bag of words and address specific features – were not effective for cross-user evaluations so were omitted, leaving relational and task specific features only. Table 2 shows good system performance on new users, even when no user specific data is available, a promising result for developing practical systems.

ATTACHMENT PREDICTION

One of the most common uses of email, sending attachments is a convenient way to transfer files and an essential part of collaborative projects. Most attachments arrive in the context of activities, such as editing papers or preparing proposals. However, in writing the email users often forget the last step of attaching the document. As a result, emails arrive without their intended attachments, only to generate a flurry of emails attempting to correct the problem. The missing attachment problem can create a serious disruption in workflow when a critical document is not received on time, especially when the sender is now out of contact.

A number of possible interface design solutions can address this problem. A system could display suggested documents to attach in a sidebar, which would be both useful and encourage the user to select a document during message composition. A more aggressive system could alert the user to a missing attachment when she sends the message, similar to alerts for missing subjects. Both interfaces require an activation method, an underlying prediction system to determine if a message requires an attachment.

Features

While attachment prediction is more dependent on the message content than reply prediction, additional features, such as relational and task specific features, are still useful for this task. As before, a user profile is constructed for each email user based on the training data that includes the number of sent and received messages from each user as well as the percentage of these messages that had an attachment. This captures patterns of attachment behavior, such as which users send a lot of attachments. Features based on these user profiles include total number of messages sent to a user, number of attachments sent to a user, conjunctions between this information and the TO/CC fields, and relevant ratios. The additional task specific features are the position of the word "attach" in the email (first or last sentence), the word "attach" conjoined with all words in close proximity, and the amount of text in the message body.

Corpus

Our attachment prediction system was evaluated on the Enron email corpus [3], a publicly available corpus with about 150 users and a 250,000 emails. The use of this dataset posed several problems. First, the publicly available dataset does not contain attachment information, which was removed during dataset construction. We obtained a database dump of the original corpus and matched it against the public corpus, adding additional header information indicating the name of the attachment (if any) in each email. Next, a large amount of attachments were of type ".html" or ".eml," which are html text alternatives and forwarded messages, not actual attachments. The label of *needs attachment* was assigned only if the message's attachment had the extension of a popular document format, such as ".doc," ".xls," ".pdf," etc. Furthermore, many forwarded emails contained attachments and it was unclear if the attachment originated from the forwarded message or from the new sender. Forwarded messages were excluded from the corpus.

Since Enron messages are real sent messages, residual artifacts were introduced by various email clients. For example, some attachment messages included an automatic list of attachments in the body of the message, such as "⟨⟨File: E&Y Memo.doc⟩⟩" or "–E&Y Memo.doc". Since a real email draft would never include these attributes, they needed to be removed. 200 random email messages were hand examined for such attributes and regular expressions were constructed to remove the offending text. This process was repeated until the randomly selected messages were free from such text.

From the resulting corpus, 15,000 messages were selected at random from sent mail folders, comprising 1,020 messages with attachments and 13,980 messages without from 144 users. This mix of messages represents real world data

Split	*System*	*Precision*	*Recall*	F_1
User	Rule Based	0.8223	0.4490	0.5808
	No Relations	0.8381	0.6647	0.7414†
	All Features	0.8301	0.6706	0.7419†
Cross-User	Rule Based	0.8223	0.4490	0.5808
	No Relations	0.8049	0.5461	0.6507$*$
	All Features	0.7981	0.5618	0.6594$*$

Table 3. Attachment prediction results on Enron mail for three systems: rule based (the word form "attach"), no relational features, and all features. Numbers are aggregate results across 10-fold cross validation. $*$ and † indicate statistical significance at $p = .01$ and $p = .001$ respectively against the baseline using McNemar's test.

so no balancing was done. Profiles were constructed for each Enron user from emails not in the evaluation corpus.

Evaluation

The problem of a missing attachment is so vexing that frustrated users have developed a Mac OS X Mail plugin that relies on the presence of the word "attach" to alert users to possible missing attachments. [3] This rule based system is used as a baseline, where a message with the words "attach," "attached," "attaching" or "attachment," is marked as *needs attachment*. Two versions of the learning based system are evaluated: one that includes all the features (39,380 features) and is the best system (All Features), and one where relation features – 72 features total – are removed (No Relations). This evaluates the contribution of relational features to the attachment prediction problem.

Each system was evaluated on the 15,000 Enron messages using 10-fold cross validation for both user and cross-user evaluations. For cross-user evaluations, users were divided into folds so emails from a single user were only used for training or testing but not both. Aggregate results are reported across the 10 folds for each split in table 3.

Results for user splits indicate that while the rule based system can identify attachment messages with high precision – messages with the word "attach" usually have attachments – it fails to find even half of the emails with attachments. In contrast, the learning system achieves slightly higher precision and much higher recall, finding two-thirds of the attachment messages. These results are significant at $p = .001$ using McNemar's test. Unlike reply prediction, relational features are not as important for attachment prediction and fail to give any noticeable improvement to the results.

Not surprisingly, the cross-user setting is more difficult. While performance of the learning system still exceeds the rule based system – which required no training data so performance remains unchanged – recall drops by more than 10 points and precision drops below the rule based system. Attachment prediction relies heavily on message content, which changes between users. Here, relational features are slightly more helpful as they are more robust to a change in user, yielding a 0.0087 increase in F_1. Unlike reply prediction which requires user labels, attachment prediction can use old emails for training, which can improve performance.

[3] *Avoid sending Mail with unattached attachments* at `http://www.macosxhints.com`

CONCLUSION

We have presented two systems for prediction problems under the rubric of Intelligent Email. Reply prediction alerts users when an email requires a response and can be used to prevent messages needing a reply from being lost in a user's mailbox. Attachment prediction can alert users when they are about to send an email missing an attachment or be used as a trigger for a document recommendation system. Both systems open the possibility to new interface design that relieves the stress of email overload.

While each problem required some additional task specific hand crafted features they both used the same underlying system. Many other prediction problems can be formulated in a similar way [1, 2, 5]. Rather than developing a unique system for each problem, a single model for email could capture all the relevant information from which features could be extracted. Additionally, the system could rely on task context for making decisions [6]. This would be particularly useful for capturing relational patterns, which are clearly useful for reply prediction and may perform well for attachment prediction with a more complex analysis. Future work will focus on more complex user models that will allow for general purpose intelligent email classification problems.

ACKNOWLEDGMENTS

Dredze was funded by a NDSEG fellowship. This material is based upon work partially supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD03001. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or the Department of Interior-National Business Center (DOI-NBC).

REFERENCES

1. Vitor R. Carvalho and William Cohen. Recommending recipients in the enron email corpus. Technical Report CMU-LTI-07-005, Carnegie Mellon University, Language Technologies Institute, 2007.
2. William W. Cohen, Vitor R. Carvalho, and Tom M. Mitchell. Learning to classify email into "speech acts". In *Empirical Methods in Natural Language Processing (EMNLP)*, 2004.
3. Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*, 2004.
4. Carman Neustaedter, A.J. Bernheim Brush, Marc A. Smith, and Danyel Fisher. The social network and relationship finder: Social sorting for email triage. In *Proc. of the Conf. on Email and Anti-Spam (CEAS)*, Mountain View, CA, 2005.
5. Richard B. Segal and Jeffrey O. Kephart. MailCat: An intelligent assistant for organizing e-mail. In *AAAI*, 1999.
6. Jianqiang Shen, Lida Li, Thomas G. Dietterich, and Jonathan L. Herlocker. A hybrid learning system for recognizing user tasks from desk activities and email messages. In *IUI*, 2006.
7. S. Whittaker, V. Bellotti, and P Moody. Introduction to this special issue on revisiting and reinventing e-mail. *Human-Computer Interaction*, 20(1-2):1–9, 2005.
8. Steve Whittaker and Candace Sidner. Email overload: exploring personal information management of email. In *Computer-Human Interaction (CHI)*, 1996.