Email Task Management: An Iterative Relational Learning Approach

Rinat Khoussainov and Nicholas Kushmerick
School of Computer Science and Informatics
University College Dublin, Ireland
{rinat, nick}@ucd.ie

Abstract

Today's email clients were designed for yesterday's email. Originally, email was merely a communication medium. Today, people engage in a variety of complex behaviours using email, such as project management, collaboration, meeting scheduling, to-do tracking, etc. Our goal is to develop automated techniques to help people manage complex activities or tasks in email. The central challenge is that most activities are distributed over multiple messages, yet email clients allow users to manipulate just isolated messages. We describe machine learning approaches to identifying tasks and relations between individual messages in a task (i.e., finding cause-response links between emails) and for semantic message analysis (i.e., extracting metadata about how messages within a task relate to the task progress). Our key innovation compared to related work is that we exploit the relational structure of these two problems. Instead of attacking them separately, in our synergistic iterative approach, relations identification is used to assist semantic analysis, and vice versa. Our experiments with real-world email corpora demonstrate an improvement compared to non-relational benchmarks.

1 Introduction and Background

A large proportion of every-day information comes to us in the form of natural language text, with email being one of the biggest sources. Many people spend significant amounts of time handling their email, and email overload is becoming a critical problem [Whittaker and Sidner, 1996].

One of the reasons for this problem is that email has transformed over the years from a communication media for simple message exchange, to a "habitat" [Ducheneaut and Bellotti, 2001] — an environment where users engage in a variety of complex *activities* or *tasks*. Examples include meeting scheduling, using email for reminders and to-do lists, e-commerce transactions, project management and collaborative work. In many cases, email serves as the primary interface to one's workplace. However, email clients offer little support for this activity oriented use of email. Email programs are still designed mainly to manipulate individual messages. As a result, their advanced features in automated email management are confined to filtering individual messages, and simple message threading.

Our goal is provide task support in email that would help users to manage their email-based activities more effectively. In many cases, such activities manifest the user's participation in various structured processes or workflows. Such processes are represented in email by *groups of related messages*, where each message can indicate some change of the process status. We are interested in being able to recognise such tasks in email, organise messages within a task, and track the task progress.

For example, a consumer purchasing from an e-commerce vendor may receive emails that describe the current status of the transaction, such as order confirmation, notification of shipment, information about shipment delays or cancellations. A manager may receive a series of messages about hiring a new candidate, such as application documents, reminders about upcoming interviews, references, and decision notifications. A single user may be involved in many such activities simultaneously.

A task-oriented email client would allow the user to manage activities rather than separate messages. For instance, one can envisage the user being able to quickly inquire about the current status of unfinished e-commerce transactions or check the outcome of recent job interviews. Some process steps could be automated, such as automatically sending reminders for earlier requests. Similarly, the email client could remind the user when her/his input is required in some activity or prominently notify him when an expected document arrives in a high-priority task.

Previous work in this area has mainly focused on two distinct problems: finding related messages and semantic message analysis. The goal of *finding related messages* is to group email messages into tasks — i.e., sets of messages reflecting the user's participation in some process or workflow and, possibly, to establish conversational links between messages within a task. Note that tasks need not correspond to threads: a task can have multiple conversation threads, and a thread can be related to several tasks. Specifically, 'In-Reply-To' headers may not reflect the correct relations between messages in a task, since users often use the 'Reply' button to start a new conversation, or neglect to use it when continuing an existing conversation. Likewise, organising emails into folders can be orthogonal to tasks. For instance, users may have a single folder for urgent messages (often the Inbox), while these messages can be from different tasks.

Semantic message analysis involves generating metadata for individual messages in a task that provides a link between the messages and the changes in the status of the underlying process, or the actions of the user in the underlying workflow. For example, a message can be described as an e-commerce order confirmation, a meeting proposal, or a delivery of information. Of course, such semantic analysis requires making assumptions regarding what actions are available to the user in an activity, what state transitions are possible in a process, etc. Notice, however, that such assumptions would not actually limit the user — they would exist only in the email client program to provide the advanced task-oriented functions.

Our work is based on the idea that related messages in a task provide a valuable context that can be used for semantic message analysis. Similarly, the activity related metadata in separate messages can provide relational clues that can be used to establish links between messages and subsequently group them into tasks. (See section 3 for more detailed motivational examples). Instead of treating these two problems separately, we propose a *synergetic iterative approach* where identifying related messages is used to assist semantic message analysis and vice versa. Our key innovation compared to related work is that we exploit the *relational structure* of these two tasks.

There has been substantial research recently on automatically finding related messages [Rhodes and Maes, 2000], sorting messages into folders [Crawford et al., 2002], and extracting a set of related messages (a task) from email given a seed message [Dredze et al., 2004]. Also, several email tools have been proposed that try to utilise this information to help users manage their email [Windograd, 1987, Bellotti et al., 2003].

For semantic message analysis, [Cohen et al., 2004] proposed machine learning methods to classify emails according to the intent of the sender, expressed in a verb-noun ontology of *"email speech acts"*. Examples of such speech acts are "Propose meeting", "Deliver information", etc. Similar problem has also been tackled in [Horvitz, 1999], where a calendar management system is described based on classifying and extracting information from emails having to do with meeting requests.

We make the following contributions: (1) We investigate several methods for identifying relations between messages and grouping emails into tasks. We use pair-wise message similarity to find potentially related messages, and hierarchical agglomerative clustering [Willet, 1988] is used to group messages into to tasks. We extend the message similarity function to take into account not only the textual similarity between messages, but also the available structured information in email, such as send dates, and message subjects. **(2)** We propose a relational learning approach [Neville and Jensen, 2000] to email task management that uses relations identification for semantic message analysis and vice versa. In particular, we investigate how (a) features of related messages in the same task can assist with classification of email speech acts, and how (b) information about message speech acts can assist with finding related messages and grouping them into tasks. Combining these two methods yields an iterative relational algorithm for speech act classification and relations identification. **(3)** We evaluate our methods on real-life email.

The rest of the paper is organised as follows: Sec. 2 describes the email corpora used in this study and introduces the *"email speech acts"*, the semantic metadata that we use; Sec. 3 presents the overall approach and formulates the specific problems that we address; Sec. 4, 5, and 6 describe our three main contributions; finally, Sec. 7 concludes the paper with a discussion and outlook at the future work.

2 Email Corpora

Although email is ubiquitous, large and realistic email corpora are rarely available for research purposes due to privacy concerns. We used two different real-world email corpora for our study. Unfortunately, the need to manually annotate emails has limited both the number of email corpora used in our study and the size of each corpus.

The first corpus contains messages that resulted from interaction between users and online e-commerce vendors [Kushmerick and Lau, 2005]. It contains messages from six vendors: half.com, eddiebauer.com, ebay.com, freshdirect.com, amazon.com and petfooddirect.com. These messages reflect the user's participation in online auctions and purchasing of goods in online shops. The corpus contains 111 messages from 39 business transactions. Examples of the messages include confirmations for online orders, shipment notifications, and auction bid status. Obviously, most of these messages were automatically generated by the vendor's software. The messages in this corpus were manually

grouped into email tasks corresponding to different transactions and annotated with the task labels.

The second corpus used in our study, the PWCALO corpus [Cohen et al., 2004], was generated during a four-day exercise conducted at SRI specifically to generate an email corpus. During this time a group of six people assumed different work roles (e.g. project leader, finance manager, researcher) and performed a number of activities.

Each email has been manually annotated with two types of labels. The first type of labelling groups messages into tasks corresponding to separate conversations and establishes the links between messages similarly to the 'In-Reply-To' header, except that in our case the links are semantic. That is, even if two messages do not have the 'In-Reply-To' header that links them (in fact, none of the messages in our corpus had such headers) or they have different 'Subject' headers, but are semantically related, they are linked together. One message becomes the cause of the other, and the latter message is a response to the former.

The second type of annotation for this corpus represents the messages semantics. In particular, each message has been annotated according to the intent of the sender, expressed in a verb-noun ontology of *"email speech acts"*. The ontology consists of verbs, such as "Propose", "Request", "Deliver", and nouns, such as "Information", "Meeting", "Data", "Opinion". The details of the annotation ontology for this email corpus are presented in [Cohen et al., 2004]. Each speech act is composed of a verb and a noun. Examples of such speech acts are "Propose meeting", "Deliver information", etc. A message may contain multiple acts. For the purposes of this study, we use only the 5 most frequent verbs, giving us 5 different speech acts: "Propose", "Request", "Deliver", "Commit", and "Amend".

To perform experiments in relational learning, we need to ensure that our training and testing sets are unrelated. So, we generated two independent email sets from the PWCALO corpus. The first corpus, called "User 1", contains all emails sent or received by one participant (160 emails in total). The second corpus, called "User 2", contains all emails sent or received by another participant but not received by the first one (33 emails in total).

3 Problem Decomposition

The idea behind our approach is that related messages in a task provide a valuable context that can be used for semantic message analysis. Similarly, the activity related metadata in separate messages can provide relational clues that can be used to establish links between emails and group them into tasks. Instead of treating these two problems separately, we propose an iterative synergetic approach based on *relational learning*, where task identification is used to assist semantic message analysis and vice versa.

1: Identify initial relations (Problem 1)
2: Generate initial speech acts (Problem 2)
loop
  3: Use related emails in the same task to clarify speech acts (Problem 3)
  4: Use speech acts to clarify relations between messages (Problem 4)
end loop

Figure 1: Iterative relational learning approach to task management.

In a non-relational approach, we would use the content of a message to assign speech act labels to this message. Similarly, we would use some content similarity between messages to decide whether these messages are related (and, hence, belong to the same task).

In a relational approach to speech act classification, we can use both the message content and features of the related messages from the same task. More specifically, we propose to use the message content and the speech acts of related messages for deciding on speech acts of the given message. For example, if a message is a response to a meeting proposal, then it is more likely to be a meeting confirmation or refusal, and this additional evidence can be combined with information obtained from the message content. Similarly, we can use messages' speech acts to improve relations identification. For example, if the suspected cause message is classified as meeting proposal and the suspected response message is classified as meeting confirmation, they are more likely to be related, than a pair where a request follows a confirmation. Therefore, we can identify the following four problems:

Problem 1: Identify relations between emails using content similarity only (i.e. without using messages' speech acts);

Problem 2: Classify messages into speech acts (semantic message analysis) using the messages' content features only (i.e. without using information about related messages in the same task);

Problem 3: Use the identified related messages to improve the quality of speech acts classification for a given message;

Problem 4: Use the messages' speech acts to improve identification of relations (links) between emails.

The outputs from these four problems can then be combined into a synergetic approach to task management based on an iterative relational classification algorithm illustrated in Figure 1. The presented algorithm can be viewed as the process of iterative re-assignment of speech act and relation labels and is similar to the *collective classification* procedure described in [Neville and Jensen, 2000].

In this paper, we investigate possible approaches to solving Problems 1, 3, and 4. For solving Problem 2, we use the standard text classification methods with bag-of-words document representations which have been evaluated already in [Cohen et al., 2004]. Finally, we combine the proposed methods into the iterative classification algorithm and evaluate its performance.

4 Finding Related Messages (Problems 1, 4)

Grouping messages into tasks. Our approach to grouping email messages into tasks is based on a hierarchical agglomerative clustering (HAC) algorithm [Willet, 1988]. A HAC algorithm requires a similarity function between messages. The algorithm starts with each message assigned to a separate cluster and then keeps merging pairs of clusters until some threshold condition is met. The two clusters merged at each step are selected as the two most similar clusters. We use the complete link method, where the similarity of two clusters is given by the smallest pair-wise similarity of messages from each cluster. To derive the stopping condition for clustering, we calculate the average pair-wise similarity for all messages in the email corpus. The clustering stops when the similarity between the two most similar clusters becomes less than the corpus-wide average similarity.

The clustering performance depends crucially on the similarity function. For text clustering, it is usually defined using the TF/IDF-based cosine similarity measure. In case of email, however, we have additional structured information, such as the message subject and send date. We propose here the following modifications to the similarity function and the clustering method itself.

Cos, S: The similarity is defined as the TF/IDF cosine similarity between message texts. However, the terms appearing in the subject are given a higher weight. The idea is that people usually try to summarise the content in the subject line and, hence, subject terms are more important.

Cos, T: The similarity is defined as the cosine text similarity with *time decay*. The idea is that messages from the same task tend to be sent around the same time. Therefore, two messages with a large send time difference are less likely to be in the same task. We use the following formula: $Sim(m_1, m_2) = Cosine\_Sim(m_1, m_2) * \exp(-\alpha * Norm\_time\_diff(m_1, m_2)$, where $Cosine\_Sim$ is the cosine message similarity, $Norm\_time\_diff(m_1, m_2)$ is the time difference between messages divided by the maximum time difference, and α is a decay parameter.

Cos, S+T: This method combines the subject-based message similarity with time decay (hopefully, combining the benefits of the two previous methods).

SC: The average similarity between messages in large email corpora is usually very small. This causes the clustering algorithm to produce a small number of large clusters with several tasks per cluster resulting in a higher task identification recall but lower precision (see below for definitions of precision and recall in this case). To address this problem, we run the HAC algorithm recursively on the clusters obtained after the top-level clustering, i.e. we sub-cluster them. Note, that the stopping criteria for each such sub-clustering is calculated within the given cluster, not corpus-wide.

SC, TRW: Because different tasks may use different terminology, it may be possible to improve sub-clustering, if we adjust the term weights to reflect their importance within the given sub-cluster. To do this, we calculate IDFs within the given top-level cluster and then multiply the corpus-wide IDFs by the cluster-specific IDFs. We call this modification *term re-weighting*.

SC+Thr, TRW: Sub-clustering all top-level clusters indiscriminately may result in some single tasks being split across multiple clusters. This would result in a drop in task identification recall (see below for definitions of precision and recall in this case). We propose to use a sub-clustering threshold to prevent this. For each top-level cluster, we calculate its "density" as the average pair-wise similarity between messages in that cluster. We then calculate the average density and only sub-cluster the top-level clusters with a smaller than average density.

We compared these various methods on both the e-commerce corpus and the free-text PWCALO email. To evaluate our algorithm, we must compare the extracted tasks with a set of human-annotated tasks. Because there may be different number of tasks in each case, we cannot simply use traditional classification performance measures evaluating the agreement between two task labellings for a given data set. Instead, we must compare two clusterings: one is generated by our clustering algorithm; the other one (the correct clustering) is defined by a human annotator who partitioned emails into tasks. To compare two clusterings, we use precision and recall measures as defined in [Kushmerick and Heß, 2003]. Consider all possible pair-wise combinations of messages in a given corpus. Let a be the number of pairs of messages that are correctly assigned to the same task (where the correct assignments are given by the human-generated clustering); let b be the number of message pairs that are incorrectly assigned to the same task; and let c be the number of message pairs that are incorrectly assigned to different tasks. The clustering precision is defined as $a/(a+b)$, recall as $a/(a+c)$, and F1 is their harmonic mean. Tables 1 and 2 present the corresponding results for different modifications to the algorithm as described above.

The results show the importance of the additional features used in the message similarity function. In particular, we can see that taking into account the time difference between

Method	Precision	Recall	F1
Cos	0.08	0.72	0.15
Cos, S	0.09	0.88	0.16
Cos, T	0.26	0.89	0.40
Cos, S+T	0.27	0.89	0.42
SC	0.85	0.62	0.72
SC, TRW	0.94	0.62	0.75
SC+Thr, TRW	0.94	0.72	0.82

Table 1: Task identification: E-commerce corpus

Method	Precision	Recall	F1
Cos	0.61	0.96	0.75
Cos, S	0.62	0.94	0.75
Cos, T	0.50	0.90	0.65
Cos, S+T	0.58	0.98	0.73
SC	0.91	0.51	0.66
SC, TRW	0.92	0.30	0.45
SC+Thr, TRW	0.92	0.73	0.81

Table 2: Task identification: Free-text corpus ("User 1" subset)

messages has a significant effect on the task identification quality on the e-commerce. As we expected, using sub-clustering improves precision on both corpora, since it results in smaller more "precise" clusters. However, it also hurts recall, because some single tasks become split across multiple clusters. Using the sub-clustering threshold helps to fix this problem on both corpora, resulting in a much better recall with only a slight loss in precision. Finally, terms re-weighting proves very effective for the e-commerce corpus, but does not work so well for the free-text email. The reason for this is that e-commerce emails contain a lot of standard automatically generated text, which is not very helpful in distinguishing between different transactions. Terms re-weighting helps to make the unique transaction data (such as transaction ID or details of purchased goods) more prominent and, thus, more useful.

Overall, our approach achieves a better performance on the e-commerce corpus than on the free-text email. While this is an indication of the fact that free text is more difficult to work with, we believe it also shows the inherent difference between the two corpora. The tasks in e-commerce email are well-structured: there is a clear understanding of how to group messages into tasks by mapping them onto the underlying business transactions. In free-text email, the definition of a task is more fuzzy, and even human annotators may disagree on the correct task partitioning.

Finding links between messages without using speech acts (Problem 1). In our PWCALO corpus, each message is a response to at most one other message (i.e. all conversations have a tree structure rather than a more generic DAG). Therefore, our algorithm for finding related messages proceeds as follows. For each email in the corpus, we find the most similar preceding (in time) email using the pair-wise message similarity function proposed in Sec. 4. We then test for a pruning condition for these two messages (as described below) and, if the pruning test fails, we establish a link between the messages.

Method	Precision	Recall	F1
Cos	0.83	0.80	0.81
Cos, T	0.84	0.80	0.82
Cos, S	0.82	0.83	0.82
Cos, S+T	0.83	0.81	0.82

Table 3: Finding relations without using speech acts ("User 1" free-text subset)

There may be multiple pairs of messages having non-zero similarity in a corpus. Clearly, not all of these messages are actually related. Therefore, we would like to be able to prune the links suggested by the similarity function. One way is to use some threshold value: if the similarity between messages is below the threshold, then we would assume that the messages are not related. Table 3 presents the quality of the relations identification for different modifications of the similarity function tested on the "User 1" subset of the PWCALO corpus.

Finding links between messages using speech acts (Problem 4). We treat the problem of finding links between messages using speech acts as a supervised learning task. We assume that we have access to a training set, which provides the correct labels for both speech acts and message relations. The goal is to use this information to improve our performance on an unseen email corpus. There are multiple ways for how the speech acts information can be taken into account when establishing links between messages. For example, we could calculate the empirical probabilities of one speech act following another in the training set, and use this information to modify the similarity function used on a test set. Since there can be numerous such modifications, we decided to rely instead on a classification algorithm to identify useful speech act patterns.

From the given labelled email corpus, we produce a set of training instances as follows. For each message in the corpus (child), we identify the most similar preceding message (parent) using the previously defined similarity function. For each such pair of messages, we create one training instance with one numeric feature for the similarity between messages, and two subsets of binary features for each possible speech act (10 features in total). The first binary subset is filled with speech acts of the parent message: 1 if the message has this speech act, 0 otherwise. The second binary subset if filled with speech acts of the child message. The class label for the instance is positive if the corresponding messages are related and negative otherwise. The resulting classifier can then be used to identify links between messages in an unseen email corpus.

To evaluate the potential for improvement from using speech acts, we tried to train and test a classifier on the

same "User 1" subset of PWCALO. Obviously, if we cannot get any improvement even on the training set, it is unlikely we will do well on an unseen test set. We use the SMO implementation of support vector machines as our classification algorithm [Platt, 1999]. We obtain precision 0.91 and recall 0.80, so that F1 is 0.85. Compared to Table 3, using speech acts were a more effective pruning method, resulting in the increase in precision with only marginal loss in recall.

5 Classifying Speech Acts (Problems 2, 3)

Classifying speech acts without using related messages (Problem 2). We treat the problem of email speech act classification as a supervised learning task. We use the standard text classification methods with bag-of-words document representations similar to [Cohen et al., 2004]. Each message is represented by a set of features, one for each term appearing in the emails. The feature value is equal to the number of occurrences of the corresponding term in the message text.

An important detail in speech acts classification is that emails may contain quoted text from previous messages. While such text fragments are usually helpful in establishing links between messages (since they make messages more similar), they can confuse a speech acts classifier. Thus, we used a simple heuristic approach to identify the quoted text so that it can be removed before speech act classification.

Classifying speech acts using related messages (Problem 3). We adopt here the relational learning terminology used in [Neville and Jensen, 2000]. Again, each email message is converted to a data instance represented by a set of features. However, unlike in the previous case (i.e. Problem 2), these features can be derived now from the content of the given message (*intrinsic features*) as well as from the properties of related messages in the same task (*extrinsic features*). The goal is to learn how to assign speech acts to emails from a training set data instances labelled with both the correct speech acts and relations. The question we are studying here is whether the extrinsic features, obtained using identification of related emails, can help in classifying messages into speech acts. That is, we want to know whether speech acts of "surrounding" messages can help in classifying speech acts of a given message.

To represent the intrinsic features of a message, we use the raw term frequencies as in Problem 2. To represent the extrinsic features of a message, we use the speech acts of related messages. As we explained earlier, in our PWCALO corpus each message can have at most one parent and possibly multiple children. Consequently, each message can be viewed as a response to its parent message and as a cause for its children messages.

In addition to looking at the immediate ancestors and descendants of a message, we can also include features from several "generations" of ancestors and descendants (e.g. parents, grandparents, children, grandchildren). For each "generation" of related ancestor and descendant messages, we use a separate set of extrinsic features with one feature per each possible speech act. The value of 1 is distributed evenly between all features corresponding to the speech acts that occur in the related messages from a given generation of ancestors or descendants. For example, if children messages have "Deliver" and "Commit" speech acts, then the features of immediate descendants for these two acts are equal to 0.5 and the other features are equal to 0. Another way would be to use binary features for each speech act: 1 if the speech act is present, 0 otherwise. However, we found that the former method works better.

The number of generations included into extrinsic features is regulated by the *depth of lookup* parameters. There are two lookup depth parameters, one for ancestor messages and one for descendant messages. For the depth of lookup 0, we use only intrinsic features. For the depth of lookup 1, we use extrinsic features for the immediate ancestor and descendants (parent and children). For the depth of lookup 2, we use extrinsic features for the immediate ancestors and descendants as well as for the ancestors of the immediate ancestors (grandparents) and the descendants of the immediate descendants (grandchildren), etc.

Results. The experiments here are based on the "User 1" corpus. We evaluate the performance of speech act classification using the human-annotated (correct) relations between messages and the correct speech acts for related messages. Essentially, we are interested in the question "Can we improve speech act classification, if we know the correct links between emails and the correct speech acts for the related messages?"

Notice that using the correct speech acts for related messages does not mean that we use the class label of an instance among its features. Each message uses only the speech acts of the related messages, but not its own speech acts. Of course, in a practical iterative classification method both the relations and the speech acts will have to be initialised using automatic methods (Problems 1, 2). The experiments here are used simply to confirm that improvements are possible in principle.

For each speech act, we produce a separate binary classification problem where the goal is to identify whether the message has this act or not. The resulting data sets for classification can be highly imbalanced with a significant proportion of messages belonging to the same class (especially for rare or very frequent speech acts). Classification accuracy is not a good measure of performance on such data sets, so (following [Cohen et al., 2004]) we use the Kappa statistics instead. We use the SMO imple-

Ancest./Descend. lookup depth	0/0	0/1	1/0	1/1
Amend	0.40	0.36	0.45	0.40
Commit	0.17	0.20	0.28	0.37 v
Deliver	0.22	0.29	0.21	0.29
Propose	0.08	0.11	0.13	0.15
Request	0.09	0.17	0.14	0.31 v

Table 4: Classifying speech acts using related emails ("User 1" free-text subset)

mentation of support vector machines as our classification algorithm [Platt, 1999] and the results are obtained in 5-fold cross-validations repeated 10 times for statistical significance testing (paired T-test). Table 4 shows the resulting Kappa values for different combinations of the lookup depth for ancestor/descendant messages. The statistically significant improvements over the *base line* (both lookup depths are zero) are marked with "v" (at the 0.05 confidence level). Lookup depths above 1 are not shown due to the lack of space.

The results show statistically significant improvements for some speech acts ("Request", "Commit") and overall increases in the Kappa values. Of course, in a practical iterative classification method both the relations and the speech acts will have to be initialised using automatic methods (Problems 1, 2) and this is likely to affect the performance.

6 Iterative Algorithm for Task Management

The results in the previous sections demonstrated that the proposed methods for solving Problems 1, 2, 3 and 4 as identified in Sec. 3 show some promise for improving performance of both relations identification and speech acts classification. However, the numbers were obtained assuming the knowledge of the correct (annotated) relations and/or speech acts in the test corpus. In this section, we put these separate methods together into an iterative relational algorithm and evaluate its performance in realistic settings, when the test corpus is unlabelled. The resulting algorithm is shown in Figure 2.

The SMO algorithm was used for all classifiers. To obtain confidence scores for SMO, we used the distance from the hyper-plane normalised over all test instances. We use the similarity function with time decay (method **Cos, S+T** in Sec. 4) and threshold-based pruning to identify the initial links between messages (Problem 1).

Results. In our experiments with this algorithm, we used the "User 1" subset of PWCALO for the training set and the "User 2" subset for the test set. On each iteration, we repeated the inner speech act classification loop 10 times ($K = 10$). Figure 3 shows how the speech acts classification performance was changing in the first iteration (note the difference between the main iterations and sub-

for all speech acts a **do**
  Train classifier C_a on the training set to classify speech act a using only email content (intrinsic features)
  Train classifier R_a on the training set to classify speech act a using content and related emails (intrinsic+extrinsic features)
end for
Train classifier L on the training set to classify email links
/ Problem 1 */*
Set relations in the test set using similarity function
/ Problem 2 */*
Use classifiers C_a to set speech acts in the test set
/ Iterative classification */*
for $Iteration = 1 \ldots I$ **do**
  / Problem 3 */*
  $Theshold = 1$
  for $Subiteration = 1 \ldots K$ **do**
    for all messages m in the test set **do**
      for all speech acts a **do**
        Obtain confidence for "m has a" using R_a
        Obtain confidence for "m has no a" using R_a
      end for
    end for
    For all cases where confidence for "m has/has no a" is greater than $Threshold$ update speech acts of m accordingly
    $Threshold = Threshold/2$
    Evaluate performance for speech acts
  end for
  / Problem 4 */*
  Use L to find links between emails in the test set, update the links accordingly
  Evaluate performance for relations
end for

Figure 2: Iterative algorithm for task management.

iterations for speech acts classification!). The performance for almost all speech acts improved considerably (except for "Amend"). The initial links identification resulted in precision=recall=F1=0.95. It improved after the first iteration to precision=1.0; recall=0.95; F1=0.98, and remained the same after the second and subsequent iterations (which is understandable, since speech acts mainly improve the links precision, and the precision after the first iteration was already 1). The performance of the speech acts classification in the second and subsequent iterations remained unchanged as well. That is, the iterative classification process converged very rapidly in this case.

7 Discussion

In addition to "mere" communication, email has become a ubiquitous channel for distributing information among collaborators, workflow participants, or even for sending to-do items to oneself. Unfortunately, today's email clients offer very little high-level support for such complex behaviours.

Developing such technologies is both a useful end in itself, and a major challenge for artificial intelligence. We have

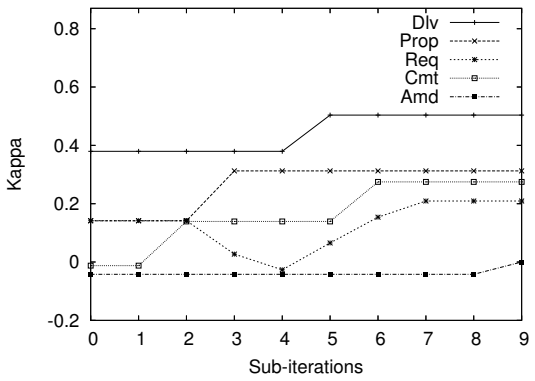

Figure 3: Speech acts classification, Iteration 1

described a variety of machine learning methods for identifying relations between emails and for analysis of message semantics. The key idea is that we synergetically exploit the relational structure of these two tasks.

We proposed an iterative relational classification algorithm for email task management that uses relations identification for semantic message analysis and vice versa. The proposed approach depends crucially on whether we can improve semantic message analysis by using the knowledge of the email links structure; and whether we improve links identification by using semantic metadata in messages. We proposed methods for solving each of these problems and evaluated their potential effectiveness. Finally, we combined these methods into a single iterative algorithm and tested it on a real-world email corpus.

Our experiments demonstrate that: **(1)** Structured features in email, such as message subject and send dates, can be very useful for identification of related messages. **(2)** The properties of related messages in the same task can be used to improve the message semantic analysis. In particular, the features of related messages in a task can improve the performance of the email speech acts classification; **(3)** The semantic metadata in messages can be used to improve the quality of relations identification. In particular, taking into account speech acts of messages helps to improve identification of links between emails.

Finally, our combined iterative classification algorithm was able to simultaneously improve performance on both speech acts and message relations. These results provide an empirical evidence in favour of the proposed synergetic approach to email task management.

There are numerous directions for future work. The presented experiments are obviously quite limited and small-scale. Therefore, experiments with different and larger email corpora can be used to better evaluate quantitatively the achievable improvements in performance. Such experiments could also allow us to study the transferability of the classifiers between email corpora and the degree to which speech act and link patterns are corpus and person-independent. However, this work requires more annotated email data, and this is one of the key challenges for the email research community at present.

Acknowledgements: This research was funded by the Science Foundation Ireland and the US Office of Naval Research.

References

[Bellotti et al., 2003] Bellotti, V., Ducheneaut, N., Howard, M., and Smith, I. (2003). Taking email to task: the design and evaluation of a task management centered email tool. In *Proc. Conf. Human Factors in Computing Systems*.

[Cohen et al., 2004] Cohen, W., Carvalho, V., and Mitchell, T. (2004). Learning to classify email into “speech acts”. In *Empirical Methods in Natural Language Processing*.

[Crawford et al., 2002] Crawford, E., Kay, J., , and McCreath, E. (2002). An intelligent interface for sorting electronic mail. In *Proc. Intelligent User Interfaces*.

[Dredze et al., 2004] Dredze, M., Stylos, J., Lau, T., Kellogg, W., Danis, C., and Kushmerick, N. (2004). Taxie: Automatically identifying tasks in email. In *Unpublished manuscript*.

[Ducheneaut and Bellotti, 2001] Ducheneaut, N. and Bellotti, V. (2001). Email as habitat: An exploration of embedded personal information management. *ACM Interactions*, 8(1).

[Horvitz, 1999] Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In *Proc. Conf. Human Factors in Computing Systems*.

[Kushmerick and Heß, 2003] Kushmerick, N. and Heß, A. (2003). Learning to attach semantic metadata to web services. In *Proc. Int. Semantic Web Conf.*

[Kushmerick and Lau, 2005] Kushmerick, N. and Lau, T. (2005). Automated email activity management: An unsupervised learning approach. In *Proc. Int. Conf. Intelligent User Interfaces*.

[Neville and Jensen, 2000] Neville, J. and Jensen, D. (2000). Iterative classification in relational data. In *AAAI Workshop on Learning Statistical Models from Relational Data*.

[Platt, 1999] Platt, J. C. (1999). *Fast Training of Support Vector Machines using Sequential Minimal Optimization*, chapter 12. MIT Press.

[Rhodes and Maes, 2000] Rhodes, B. and Maes, P. (2000). Just-in-time information retrieval agents. *IBM Systems Journal*, 39(3).

[Whittaker and Sidner, 1996] Whittaker, S. and Sidner, C. (1996). Email overloading: Exploring personal information management of email. In *Proc. Conf. Human Factors in Computing Systems*.

[Willet, 1988] Willet, P. (1988). Recent trends in hierarchical document clustering. *Information Processing and Management*, 24.

[Windograd, 1987] Windograd, T. (1987). A language/action perspective on the design of cooperative work. *Human-Computer Interactions*, 3(1).