Inferring Users' Projects from their Workstation Contents

Tom M. Mitchell, Sophie H. Wang, Yifen Huang, Adam Cheyer*

November, 2005

CMU-CALD-05-113

School of Computer Science

Carnegie-Mellon University

Pittsburgh, PA 15213

*SRI International

Melon Park, CA 94025

ABSTRACT

One key to providing intelligent assistance to workstation users is to construct machine-understandable descriptions of the user's ongoing projects, or activities, (e.g., their committee memberships, writing projects, conference organization activities), and indices describing which emails, meetings, and colleagues relate to which activity. This paper presents a program, ActivityExtractor, which examines the user's workstation contents to infer such activity descriptions. In earlier work [Huang, et al., 2004] we described an algorithm which infers activities by clustering the user's emails based on their word distributions. Here we extend this approach in several ways: (1) by incorporating a social network analysis of email senders and recipients, (2) by considering workstation contents beyond email, including contents accessible via Google Desktop Search, and (3) by allowing simple user input in the form of a list of activity/project names. We describe the ActivityExtractor algorithms and report on experiments applying these to several users' workstations.

Keywords
Email analysis, Workstation content analysis, Desktop Search, ActivityExtractor, clustering, social network analysis, machine learning.

1. INTRODUCTION

The research reported here is motivated by our goal to build intelligent workstation assistants to help users organize their workstations contents, to automate routine tasks such as email processing, and to provide other active assistance such as intelligently deciding when it is valuable to interrupt the user to suggest they read a specific email.

One essential component to providing such intelligent assistance is to have a machine-understandable description of the ongoing *projects*, or *activities*, in which the user is involved (e.g., courses they are teaching, committee memberships). Workstation users rarely maintain an explicit list of these projects in a form interpretable by the workstation, yet it is often these projects that primarily determine workstation structures such as email folders, directory-subdirectory structure, and lists of contacts.

The goal of the research reported here is to automatically infer computer-interpretable descriptions of a workstation user's projects, to support two uses:

1. To index the workstation contents by these inferred projects, thereby allowing the user to browse their emails, contacts, meetings, and files by project.
2. To use these project descriptions as the basis for knowledge-based assistance to the user. For example, given knowledge of the users' projects, a computer assistant may reason that when a new unseen email arrives and is associated with the same project as a meeting the user is about to attend, then it is useful to interrupt the user to alert them to this new email.

This paper focuses on algorithms for automatically inferring project descriptions from user workstation contents. Below we briefly summarize related research, then describe our algorithms for inferring project descriptions, and report on experiments applying these algorithms to contents from several workstations.

2. RELATED WORK

A variety of researchers have explored the use of social network analysis with email, even though most popular email applications are still oriented toward manipulating individual messages. Historically, the most popular approach to social network analysis involving email is to construct a graph using the email header information. Here, the vertices are senders or recipients of email messages, and the links denote a direct email between the nodes they connect. Graph algorithms are then employed to find the communities embedded in the graph (e.g., [Tyler, 2003]). One end-to-end system [Culotta 2004] extracts a user's social network and then extracts its members' contact information given the user's email inbox. The system identifies unique people in email, finds their Web presence, and automatically fills the fields of a contact address book using information extraction algorithms trained using conditional random fields. Another system, EmailNet [EmailNet 2004] automatically mines email traffic across the entire organization and generates social networks capturing sender-receivers information to help social scientists and organizational archivists study email communication across the organization. Another approach to help people manage email more effectively is presented by the ReMail system [Rohall 2004], which explores visualization techniques for displaying message threads and extracts important dates and message summaries. However, only structured email data, such as sender-receivers, dates and time, have been used in these systems. The unstructured text of email body is not typically utilized in conjunction with social network analysis of email.

On the other hand, automated text analysis for email has also been considered by many researchers, usually independent of social network analysis. For example, machine learning approaches to automated email classification have been applied to tasks such as detecting spam [Sahami 1998], predicting where to store a message [Segal 2000], classifying the intent of the email sender [Cohen 2004], and grouping similar messages [Mock 2001]. McCallum [McCallum 2004] analyzed email collections with an algorithm that forms "author-recipient-topic" models, which characterize the distribution of words sent by specific authors to specific recipients. The learned word distributions are represented by a mixture of multinomials (topics) which may be shared across senders and recipients. This work was later extended to model multiple roles of the sender in the role-author-recipient-topic model [McCallum 2005]. Kushmerick's activity management system [Kushmerick, 2004] works on the problem of learning workflow or process models from example execution logs. This work focuses mainly on inducing workflow process descriptions, but also employs text classification and clustering techniques to attach meaningful labels, which are available directly from an execution log and raw email messages. Some researchers have focused on learning activity-centric collaboration through Peer-to-Peer shared objects [Geyer, 2003]. Some have also worked on automatic discovery of personal topics to organize email [Surendran, 2005].

This paper builds directly on our earlier research reported in [Huang, et al., 2004]. There we described an algorithm which infers projects by clustering the user's emails based on the distribution of words within the body of the email, then applies a variety of information extraction, text classification, and statistical summarization methods to each cluster to form an explicit, structured description of the activity/project associated with this cluster. In the current paper we report on extensions to this approach along several dimensions: (1) incorporating a social network analysis of email senders and recipients together with clustering based on the email body, to infer project clusters of emails, (2) considering workstation contents beyond email, including contents accessible via Google Desktop Search, and (3) allowing simple user input in the form of a list of project names.

3. SYSTEM ARCHITECTURE

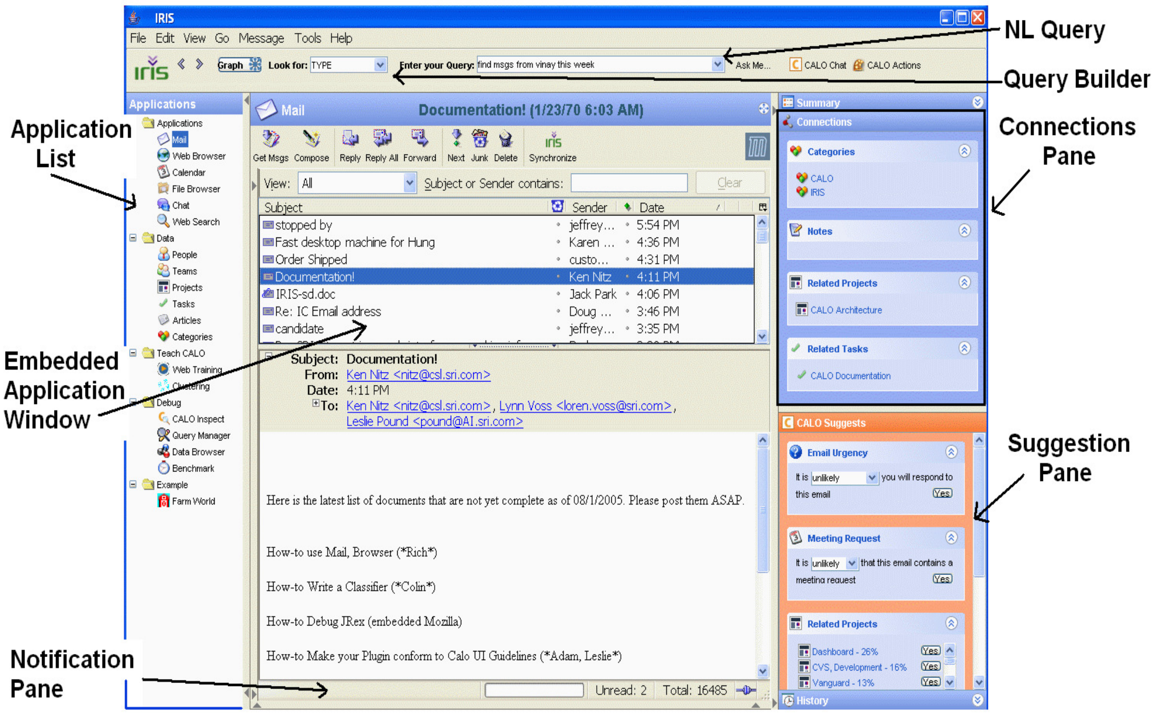
Our approach to inferring users' projects from their workstation contents is implemented in the ActivityExtractor system. ActivityExtractor consists of three central components which (a) cluster emails based on analysis of their text content, (b) perform social network analysis on each cluster to potentially subdivide it into subclusters, and (c) construct a structured representation of each cluster (project). ActivityExtractor also outputs an email classifier which can be used to incrementally assign future emails to clusters, or activities.

An example project description produced automatically by ActivityExtractor is shown in Figure 1. This description is the verbatim output of ActivityExtractor, except that we have removed person names from the list of keywords to improve readability, and we have removed phone numbers from the meeting descriptions for privacy reasons. The activity described here is a research project of the workstation user. The project name was selected by ActivityExtractor from a list of 23 project names supplied in advance by the user. Among the 245 emails that ActivityExtractor placed in this cluster, 240 were judged by the workstation user to indeed be relevant to their "CALO ActivityExtractor" project. Note the project description includes a list of emails, meetings from the online calendar, and person names from a list of contacts. Thus, the activity description serves as a basis for bringing together related information from disparate parts of the workstation.

Cluster3.0 (245 emails)

- *Project Name*: CALO ActivityExtractor
- *Keywords (omitting person names)*: ActivityExtractor, TFC, IRIS, clustering, heads, emails, collected, clusters, SRI, volunteers, CVS, ISI, stretch, organize, mobile, …
- *Primary Senders*: tom.mitchell@cmu.edu (75), sophie.wang@cs.cmu.edu (20), adam.cheyer@sri.com (16), perrault@ai.sri.com (14), tgd@eecs.oregonstate.edu (13), william.cohen@cs.cmu.edu (10), pereira@cis.upenn.edu (9), lpk@csail.mit.edu (8), mccallum@cs.umass.edu (7), …
- *Primary Recipients:* tom.mitchell@cmu.edu (94), adam.cheyer@sri.com (40), william.cohen@cs.cmu.edu (35), perrault@ai.sri.com (19), sophie.wang@cs.cmu.edu (18), ang@cs.stanford.edu (16), calo-discuss@calo.sri.com (16), tgd@eecs.oregonstate.edu (15), etzioni@cs.washington.edu (13), colin.evans@sri.com (13), ...
- *Person Names:* Adam Cheyer (0.49), Ray Perrault (0.36), Hung Bui (0.32), Melissa Beers (0.30), James Arnold (0.28), Jack Park (0.26), Sophie Wang (0.25), Tomas Uribe (0.25), jeanne ledbetter (0.24), Leslie Pack Kaelbling (0.24),...
- *Meetings*: CALO TFC telecon (0.59), CALO phone call (0.55), SRI Meeting – Chicago (0.48), SRI TFC Telecon and quarterly rpt (0.47), SRI visit. Bill and Ray. Call Melissa Beers when arriving (0.47), CALO annual mtg at SRI (0.45), CALO TFC Teleconference (0.38), Sophie, SRI phone call (0.38),...
- *User Activity Fraction*: 245/2822=0.086 of total emails
- *Intensity of User Involvement*: user authored 30% of email; (default 31%)
- *Extracted Names*: Tom(100), Adam(42), Sophie(29), Cheers Tom(27), Tom Mitchell(24), Thomas(14), Andrew(11), Adam Cheyer(9), Allen(9), William(8), Bruce(7), Jim(7), Ray(6), Vivid Focus(6), Leslie(6), Tom Dietterich(5), David(4),...
- *Extracted Dates*: 2005(19), today(16), tomorrow(11), Friday(10), May 1(8), Monday(6), 3/26/05(5), Thu(5), Wed(4), 21 Apr 2005(4), Fri(4), yesterday(4), April 11(4), Monday 4/4/05 - 9(3), 5.1(3), friday(3), Apr 25(3), weekend(3), April 27(3), April 22(3), April 15(3), 22 Apr 2005(3), January 1(2), 4/6/05(2) ...
- *Extracted Times*: morning(11), 9am(5), 8(3), 9:18 PM(2), 10:27 AM(2), afternoon(2), 9:36 AM(2), 9:44 PM(2), this morning(2), 1:10 PM(2), 10am(2)...
- *Request Emails (from user)*: <1766>, <2280>, <2399>, <2405>, <2408>, ...
- *Request Emails (not from user)*: <1006>, <1007>, <1012>, <1071>, ...

Figure 1: A project description created automatically by ActivityExtractor. The project name has been selected from a list of 23 names input in advance by the user. Integer numbers in parentheses indicate number of occurrences of the entity within the 245 emails contained in the cluster. Real-valued numbers in parentheses indicate confidences that the corresponding person or meeting is associated with this activity. Note there are some errors in the extracted names, dates, and times, due to the fact that these were extracted automatically from the bodies of the emails.



Mail — Documentation! (1/23/70 6:03 AM)

View: All   Subject or Sender contains: Clear

```
Subject                          Sender       Date
stopped by                       jeffrey...   5:54 PM
Fast desktop machine for Hung    Karen ...    4:36 PM
Order Shipped                    custo...     4:31 PM
Documentation!                   Ken Nitz     4:11 PM
IRIS-sd.doc                      Jack Park    4:06 PM
Re: IC Email address             Doug ...     3:46 PM
candidate                        jeffrey...   3:35 PM
```

Subject: Documentation!
From: Ken Nitz <nitz@csl.sri.com>
Date: 4:11 PM
To: Ken Nitz <nitz@csl.sri.com>, Lynn Voss <loren.voss@sri.com>, Leslie Pound <pound@AI.sri.com>

Here is the latest list of documents that are not yet complete as of 08/1/2005. Please post them ASAP.

How-to use Mail, Browser (*Rich*)

How-to Write a Classifier (*Colin*)

How-to Debug JRex (embedded Mozilla)

How-to Make your Plugin conform to Calo UI Guidelines (*Adam, Leslie*)

Unread: 2 Total: 16485

Email Urgency: It is unlikely you will respond to this email [Yes]

Meeting Request: It is unlikely that this email contains a meeting request [Yes]

Related Projects:
Dashboard - 26% [Yes]
CVS, Development - 16% [Yes]
Vanguard - 13% [Yes]

Figure 2. User interface of IRIS, the "CALO Intelligent Workstation". As a user works with Email, CALO suggests connections and actions that they user may wish to act upon

4. CALO INTELLIGENT WORKSTATION

ActivityExtractor has been developed as part of the CALO project, one of two projects funded under DARPA's "Personalized Assistant that Learns" (PAL) program. The goal of the PAL program is to develop an enduring personal assistant that lives with, learns from, and supports users in managing the complexities of their daily work lives. In CALO, learning happens "in vivo", inside an ever-evolving agent that can observe, comprehend, reason, anticipate, act, and communicate.

As an assistant, the more CALO understands the dynamics of the user's work life, the more useful and relevant it can be. Towards this end, the CALO research effort has developed an extensible "intelligent workstation" called IRIS [Cheyer, 2005]. As the user works with files, calendar appointments, instant messaging, the web, and email (Figure 2), the content accessed by the user is harvested and normalized into a unified, searchable data model. Events and data representing the user's activity are made available to CALO, which hypothesizes about the structure of the information it sees: what are the projects the user works on; who are the key participants in various roles for these projects; how does accessed information relate to people, projects, and tasks in the user's life; what are the relative priorities of specific tasks, messages, documents, people, and meetings? IRIS' plug-and-play learning framework and its integration of mature, third-party office applications such as Mozilla Mail, OpenOffice Calendar, and Jabber Instant Messaging set it apart from other semantic desktop systems such as MIT's Haystack [Karger, 2005], JetBrain's Omea [Jetbrains], and OSAF's Chandler [Kapor, 2005].

In CALO, "projects" play a central role, acting as a hub that ties information together -- people, teams, email, calendar appointments, files, chat messages, and tasks can all be directly linked to a project. When a new user installs IRIS, after registering calendar and mail accounts, the first recommended setup task is to ask CALO to automatically infer the user's main projects using ActivityExtractor. ActivityExtractor examines the workstation data and computes Project Descriptions such as the one shown in Figure 1. Through a simple interface, the user can then refine the automatically generated project list, removing participants or emails that are not relevant, adapting the label of the clusters, or requesting a different number of clusters to be generated. Once the user is satisfied, IRIS creates new projects or tasks from the clusters, linking all participants to their respective projects (noting level of involvement), connecting relevant email, and summarizing the projects through a list of keywords.

Project information is also very useful to other CALO learning algorithms. For example, a number of classifiers are trained using the generated project information, including an email-to-project predictor, a calendar-to-project predictor (using participants, title, and notes of an appointment), file-to-project (using author and text content of file), and chat-to-project (using participants and conversation text). This enables most information a user interacts with to be linked together through a project. Another capability that makes use of learned project information is when CALO proposes a "PrepPack" for a meeting or a person of interest; that is, an automatically constructed collection of email, web pages, files, and other prioritized content related to that meeting or person. The PrepPack algorithm selects resources by using project relationships in addition to textual similarity, recency, and popularity of access. As ActivityExtractor's algorithm becomes more accurate, the user's overall experience with CALO reaps the benefits.

5. CAPTURING AND REPRESENTING WORKSTATION-WIDE INFORMATION

Whereas earlier work such as [Huang 2004] uses only email to infer descriptions of users' projects, evidence for these projects exists across many parts of the workstation. For example, user projects are reflected in online calendars, file contents, directory hierarchies and names, and visited web pages. Fortunately, desktop search engines such as Google Desktop Search (GDS) make it feasible to efficiently search and retrieve this kind of information from across the entire workstation.

In this section we describe an approach to automatically constructing descriptions of projects, meetings, people, emails, and email addresses, representing each such entity in terms of an appropriate *word distribution*. Put another way, each entity is represented by a high-dimensional feature vector describing a word distribution, where the ith feature corresponds to the ith word in the vocabulary of words and other tokens found on the workstation. In the experiments reported here, this vocabulary contains between 20,000 and 50,000 tokens, and hence the feature vectors describing the word distributions for each entity contain tens of thousands of features.

The exact algorithm for generating word distributions varies by the type of entity, but in all cases it is designed to capture the semantics of the entity based on the current workstation contents, in a form that can be compared across all entities. For example, the word distribution associated with an email is computed from the words appearing in the header and body of the email, whereas the word distribution associated with a person name is computed from words that co-occur with this person name across the entire workstation. Below we describe how each type of entity is assigned a word distribution. Section 6 describes how these word distributions are used in the ActivityExtractor algorithm.

We first describe the algorithm for associating word vectors with emails and with email addresses:

- <u>Emails</u>: the word distribution for an email is based on the counts of the words found in the email subject line and body, with functional tokens such as "Re:" and "Fwd:" removed from the subject line. The value of the ith feature in the word distribution vector is simply the count of the number of times the ith vocabulary word appears in these fields of the email.
- <u>Email addresses</u>: the word distribution assigned to an email address is the sum of the word distributions for each email sent by, and received by that email address.

The word vectors for entities of type person name, calendar meeting, and project title are all computed by employing Google Desktop Search (GDS) as a subroutine, using its API. The general strategy in these cases is to first construct a search query based on the entity, then construct the word distribution vector based on the returned search hits for that query. For each document that matches the query (documents can be emails, files, web pages visited, or chat messages sent and received), GDS reports the document's name, plus a snippet of text characterizing the context where the query terms were found within the document. Our program converts this set of document names and text snippets into a word distribution for the entity.

More specifically, the query for each type of entity is constructed as follows:

- Person names: the name of the person is used as a query to GDS, and the returned document names and text snippets are used to create the word distribution.
- Meetings: the text description of the meeting (e.g., "meet with Sharon and William") is used as the GDS query. If no hits are obtained with this query, then progressive truncations of the query are considered, until a search success is obtained. (e.g., first attempt the query "meet with Sharon and William", then "meet with Sharon and", then "meet with Sharon", etc.).
- Project names: the user-provided name of a project is used as a query to GDS, and the returned search results are used to form the word distribution.

One can view the word distribution vector associated with each entity as a representation that summarizes a broad set of information, or associations, about the entity obtained from different parts of the workstation. Note that because we use the same word vocabulary to define the word distribution vectors regardless of the type of entity, we can use the dot product or cosine similarity between two vectors to obtain a measure of their similarity. For example, we can use this similarity metric to compute which email address is most "similar" to a particular meeting, in terms of the word distribution vectors for each. We can also use machine learning methods to jointly cluster or classify these descriptions, regardless of the type of entity. Below we describe our algorithm for ActivityExtractor, and how it employs these descriptions to discover and describe user projects.

6. ActivityExtractor ALGORITHM

ActivityExtractor implements a parameterized family of algorithms for estimating coherent clusters of emails, meetings, person names, and email addresses that may correspond to ongoing projects, or activities of the workstation user. In addition to finding coherent clusters of these entities, it also analyzes these to build structured descriptions of each proposed activity, such as the project description shown in Figure 1.

Our earlier version of ActivityExtractor [Huang 2004] considered only the emails on the user's workstation, and clusters were formed based only on the bags of words associated with emails. The family of algorithms described here extends this earlier work in several directions:

- It can jointly cluster word distributions associated with emails, calendar meetings, person names, and email addresses, enabling it to integrate a much more broad range of information from the user's workstation
- It performs a social network analysis of email senders and recipients, to refine the clusters derived from word distributions
- It allows users to input Project names, which are then used to guide the formation of clusters.

ActivityExtractor takes three inputs: (1) an email collection along with a set of additional entities represented by their word distribution vectors, (2) a user-specified parameter which specifies the initial number of clusters to be computed in step 1 of the algorithm, and (3) (optionally) a set of project names input by the user. Given these inputs, ActivityExtractor produces as output a set of project descriptions such as the one in Figure 1, along with a mapping from each project description to the input emails and entities associated with it. It does this using a three-step algorithm:

1. Cluster the input entities based on their associated word distributions.
2. (Optionally) use social network analysis over the email associated with each cluster, to further subdivide the cluster.
3. Apply information extraction and document classification methods to produce rich descriptions of the project associated with each cluster, and to associate additional types of entities (e.g., meetings, person names) with each cluster.

The following three subsections describe each of these algorithm steps in turn.

6.1 Step 1: Cluster the Word Vectors

Given a set of entities represented by their word distribution vector, the first step of ActivityExtractor is to cluster these entities. Before clustering the corresponding vectors, one pre-processing step is applied to the email vectors in order to assure that emails from the same thread (i.e., replies and forwards containing the same subject line) are assigned to the same cluster. In particular, each email thread is preprocessed by summing vectors for all emails within the thread, then replacing each of these emails' vectors by this sum.

The resulting collection of word distribution vectors is then clustered using an EM-based mixture of multinomials clustering algorithm, following [Nigam et al., 2000]. Here, clustering is viewed as a problem of estimating the components of a mixture distribution assumed to generate the collection of observed word vectors. Each mixture component is interpreted as the generator for one of the clusters, and is described by a Naïve Bayes model in which the vector feature values are assumed to be conditionally independent given the mixture component. An EM algorithm is applied, beginning with an initial assignment of vectors to clusters, then iteratively solving for a locally maximum a posteriori (MAP) assignment of vectors to clusters. This EM procedure iterates the following two steps until the change across iterations becomes negligible:

- E step: for each example vector *v* in the set of input vectors *E*, use the current model Θ to determine the probability distribution over the set of possible clusters *C*. That is, calculate the distribution over cluster labels, P(*C* | *v; Θ*), for each vector *v*. Let *E'* denote the set of these label distributions, for all example vectors.
- M step: retrain the model Θ using the label distributions *E'* computed in the E step. More formally, compute the new maximum likelihood estimate Θ' $= \arg\max_{\theta} P(\theta | E', \alpha)$, where α denotes the parameter vector for Dirichlet priors on Θ. The updates to the parameters are given by

  $$\theta'_{c_j} \equiv P(c_j \mid \theta') = \frac{1 + |E'_{c_j}|}{|C| + |E|} \quad \theta'_{w|c_j} \equiv P(w \mid c_j; \theta') = \frac{1 + freq_{c_j}(w)}{|V| + |V_{c_j}|}$$

  where c_j is the *j*th cluster, *w* is any word in the vocabulary V, E'_c is the probability mass associated with cluster *c* summed over all vectors, |*E*| is the number of vectors, and |*C*| is the number of clusters. Also |*V*| is the number of distinct words in the vocabulary, and $freq_{cj}(w)$ is the number of occurrences of word *w* across all vectors in class c_j, with occurrences within each vector *v* weighted fractionally by P(c_j|*v*, Θ). Similarly, $|V_{cj}|$ is the sum of $freq_{cj}(w)$ taken over all words *w* in the vocabulary *V*. Here the 1/|C| term in the first expression, and the 1/|V| term in the second expression are smoothing terms commonly used in text analysis, and correspond to setting Dirichlet priors on Θ'_{cj} and $\Theta'_{w|c}$.

One important issue in EM-based clustering algorithms is initializing the model Θ before the first iteration. While EM is guaranteed to converge to a locally maximum a posteriori estimate of Θ, the initial conditions determine which local maximum is reached. We consider two alternative approaches to initializing the model, both based on forming initial assignments of vectors to clusters, then setting the values of Θ according to the M-step equations above. These two approaches are:

1. VDI: This approach assigns five initial vectors to each cluster, attempting to maximize the average inter-cluster distance and minimize the average inner-cluster distance. Distance is measured by the cosine similarity between feature vectors of two emails. We use a heuristic method to search for these initial clusters, and consider only the words in the email subject lines in this step.
2. GDI: This approach uses the optional user input Project names (e.g., "machine learning course 701") as queries to Google Desktop Search and forms the corresponding word vectors as described in section 5. The word vectors for the various project names are used to train the initial model Θ using the M-step equations.

6.2 Step 2: Perform Social Network Analysis

The clusters formed during step 1 tend to group together entities with similar word distributions, but they are far from a perfect reflection of the user's projects, for a number of reasons. For example, the clustering algorithm must assign each vector to some cluster, including spurious unsolicited emails that are not actually related to any of the user's ongoing projects. Also, while the word distribution vectors for emails take into account subject and body words, they fail to utilize the important information in the network of email senders and recipients. To address some of these imperfections, step 2 performs a social network analysis on the senders and receivers of emails associated with each cluster output from step 1. It then uses this analysis to split each cluster into subclusters derived from the disconnected components of the social network graph.

In more detail, a social network graph is created from the emails of each cluster. For each cluster c, an un-directed graph $G_c(N,E)$ is created based on using the sender's and recipients' email addresses. Each node in N indicates an email address in either the "From", "To" or "Cc" field. An edge is added between two nodes if at least one email in the cluster contains both nodes in its header. The weight on the edge is the number of emails containing both nodes in the header. The workstation user's email address is excluded from the node set N, because their email address is obviously linked to every node in the graph.

The graph G_c is then examined to determine the number of disconnected subgraphs it contains. Each such disconnected sub-graph is used to define a coherent subcommunity of email addresses (the nodes within this cluster). The output of Step 2 is a set of subclusters for each initial cluster c, where each subcluster $S_{c,i}$ contains all emails associated with the *i*th disconnected subgraph within G_c. The net effect of this subclustering is to separate out different subcommunities of email addresses, which often reflect different projects of the user.

6.3 Step 3: Create Final Project Descriptions

Once it has produced the final clusters from step 2, ActivityExtractor constructs a representation of each cluster (project) by extracting information from the emails associated with the cluster, and also by associating calendar meetings and person names to each cluster. In particular, it extracts (1) keywords based on a chi-squared test of the relevance of each word to the cluster, (2) the list of senders and recipients of email within the cluster, rank ordered by the volume of mail they sent and received (3) the fraction of user email that falls into this cluster, as a measure of its importance, (4) the fraction of email within the cluster that is authored by the user, as a measure of the user's engagement in this project, (5) person names, dates, and times extracted from the cluster emails, using the Minor Third information extraction package [Cohen 2004], and (6) a classification of some emails as "requests for action", using an email classifier previously trained with the MinorThird package [Cohen 2004]. In addition to this information extracted from emails, it also assigns meetings from the online calendar, and person names from a contacts list to each activity. These assignments are based on the word distribution vectors assigned to the meetings and person names, as described above in section 5. The degree of association of a meeting or person name with a particular project is computed based on the similarity of its word vector to the sum of word vectors of all emails in the project cluster.

7. EXPERIMENTAL EVALUATION

This section summarizes results from ActivityExtractor applied to three of the authors' workstations. Our experimental evaluation of ActivityExtractor was driven by three primary questions: First, what is the general quality of the clusters and the activity descriptions produced by ActivityExtractor? Second, what is the impact of using information from the entire workstation, as captured by Google Desktop Search? Third, what is the impact of incorporating the social network analysis step, in contrast to our earlier approach of clustering based only on the distribution of words in the email?

We evaluated ActivityExtractor using three workstations that are in routine use. Note we could not evaluate ActivityExtractor on isolated email collections, but required full workstations, because it employs information from the entire workstation via Google Desktop Search.

- Workstation A. This workstation is used routinely by a university professor. The email collection contains essentially all emails sent and received over a period of 45 days, minus spam emails. This collection contained 2822 emails, which were not organized into folders. This workstation has an online calendar containing 1231 meetings, and no chat information. We also extracted 2159 email addresses and 470 person names from the email headers. Finally, the user contributed 23 project names reflecting projects they were involved in, such as "faculty hiring and interview," "machine learning course 701," and "CALO ActivityExtractor."
- Workstation B. This workstation is used routinely by a university staff researcher. The email collection contains 420 emails, organized into 8 folders. This workstation has no online calendar, and no chat information.
- Workstation C. This workstation is used routinely by a university graduate student. The email collection contains 617 emails, which were organized into 11 folders. This workstation has no online calendar, and no chat information.

7.1 Quality of Clusters

In order to test the overall quality of the email clusters output by ActivityExtractor, we took advantage of the pre-existing email folders on workstations B and C to examine how accurately ActivityExtractor would reconstruct these folders given just the emails and the target number of clusters/folders. In this evaluation, we ran only Step 1 of the ActivityExtractor algorithm, and used no social network analysis. However, we did test both the VDI and GDI initialization methods. Recall that GDI uses desktop search to create word distribution vectors for each user-provided project name, then initializes clusters using these vectors. In this case, the owners of workstations B and C contributed short project name queries based on their folder names. In contrast, VDI has no access to project names or desktop search. The results, displayed in Table 1, give the average of the cluster accuracies for both workstations and both initialization methods. The accuracy of a clustering is computed by first mapping each cluster to its most similar folder, then measuring the fraction of cluster emails that actually belong to this folder. Note the final row in the table gives the average cluster accuracies obtained by randomly assigning emails to folders. Notice that while both VDI and GDI produce significantly better-than-random reconstructions of the folder clusters, the accuracies are far from perfect. This is in part due to the fact that folder assignments are themselves somewhat arbitrary, but it is also partly due to many errors in assigning unrelated emails to the same cluster. Interestingly, despite the fact that clusters may be impure, and contain some emails unrelated to the primary cluster theme, the resulting project descriptions are often nevertheless of good quality. This is apparently due to the fact that the final project descriptions are computed from the dominant statistics of the clusters, and are therefore somewhat immune to small numbers of unrelated emails.

	Workstation B	Workstation C
VDI	0.48	0.41
GDI	0.66	0.58
random	0.13	0.09

Table 1. Impact of using desktop search to initialize project clusters.

Average of cluster accuracies when using VDI versus GDI initialization of clusters. GDI uses Google Desktop Search and input project names, whereas VDI has access to neither. Accuracies are from Step 1 clustering, before social network analysis. "Random" shows the expected accuracy when emails are randomly assigned to clusters.

7.1 Impact of Google Desktop Search

We also examined the impact of incorporating workstation-wide information, in the form of word vectors calculated using Google Desktop Search, in two ways. First, we considered the impact of using GDI initialization, in which user-supplied project names were assigned word vectors using Google Desktop Search, and then used to initialize clusters. The results are apparent by comparing the GDI row of Table 1 to the VDI row. As is clear from this table, the information provided by these word vectors leads to significantly higher accuracies in folder reconstruction.

A second way in which word vectors from Google Desktop Search are incorporated into ActivityExtractor is in the algorithm for assigning meetings and contact person names to projects, discussed in Section 6.3. We evaluated the accuracy with which meetings and person names were assigned to projects as follows: for each project, we considered the five meetings assigned to the project with strongest confidence, and the five most confident person names. In the event that the user felt fewer than five meetings or people were actually associated with that project, then we considered only the total number given by the user. For the 23 projects in Workstation A, the total number of meetings considered was therefore 100, and the total number of person names was 110. Among these meetings and persons, we then determined the fraction of meetings and person names correctly assigned to the project. The results are summarized in Table 2.

	Workstation A
Meeting accuracy	0.75
Person Name accuracy	0.89

Table 2. Accuracy of assignment of Meetings and Person Names to projects, for Workstation A.

As is clear from this table, the assignment of meetings and person names to projects, based on word vectors calculated using Google Desktop Search, is quite accurate.

7.2 Impact of Social Network Analysis

To test the impact of social network analysis (SNA), we again examined the average cluster accuracies, comparing accuracies of clusters obtained before and after the social network analysis step (i.e., the clusters output from step 1 versus step 2). Because SNA splits clusters into multiple subclusters, hopefully separating the email into subclusters associated with different folders, we chose to measure the post-SNA accuracy by considering only the cluster accuracy of the *largest* subcluster produced from each input cluster. This captures the effect of SNA by measuring how 'pure' the largest post-SNA subcluster is.

The results are summarized in Table 3. While Workstation A had no predefined email folders, the owner of the workstation contributed a list of 23 project names, and evaluated the accuracy of the clusters by hand labeling these. As Table 3 makes clear, the impact of social network analysis is to successfully separate the initial clusters into more pure subclusters.

	Worksta A	Worksta B	Worksta C
Before	na	0.66	0.58
After	0.80	0.83	0.73

Table 3. Impact of social network analysis on accuracy.

All results employ GDI initialization. "Before" indicates average cluster accuracy before Social Network analysis, after Step 1 clustering. "After" indicates the average of the accuracies of the largest subcluster from each step 1 cluster, following social network analysis.

Another view of the impact of social network analysis is provided by examining the topics of the clusters output from Step 1 of ActivityExtractor, along with the topics of the subclusters output by the Step 2 social network analysis. This hierarchy of clusters and subclusters for Workstation B is shown in Table 4. Here the cluster and subcluster names were contributed by the workstation user, after examining the emails within the (sub) cluster. Alongside each cluster and subcluster is the fraction of its emails that belong to the dominant cluster topic. For example, the dominant topic of the first cluster is "Helpdesk." It contains 28 emails, of which 9 are actually from the folder "Helpdesk." The three subclusters below this are the result of social network analysis. In this case, one subcluster is now a pure (9/9) collection of "Helpdesk" emails. The other two subclusters are not directly about Helpdesk, and have been successfully split off by the social network analysis into more pure collections about other topics. Notice for every initial cluster in this data set, the effect of SNA is to produce a largest subcluster that more purely captures the dominant topic of the initial cluster. Subclusters containing only a single email are omitted for the sake of brevity.

- Helpdesk (9/28)
    - Helpdesk (9/9)
    - CMU account set up (5/9)
    - CALO account set up (9/9)
- Meetings (13/57)
    - Meetings (13/28)
    - Family (7/7)
    - Friends (7/7)
    - Officemate (3/3)
- Castp (50/53)
    - Castp (50/51)
- Hdu (29/32)
    - Hdu (29/29)
- Idiap (34/39)
    - Idiap (34/34)
- CALD (14/24)
    - CALD (14/18)
- Scs (10/39)
    - Scs (10/20)
    - Morris (4/4)
    - Text-learning (8/8)
- Work (132/178)
    - Work(132/147)

Table 4. Impact of social network analysis – clusters and subclusters.

Cluster names were contributed by the workstation user, and describe the dominant topic of cluster emails. Numbers in parentheses indicate the proportion of emails belonging to that dominant topic. Subclusters containing only 1 email have been omitted for brevity.

8. CONCLUSION

We have presented an approach to discovering projects, or activities, of a workstation user by automated analysis of workstation contents. In contrast to earlier approaches, we combine a text analysis with social network analysis of email, and with workstation-wide information obtained through desktop search. Experimental results over three workstations indicate that (1) quite accurate structured descriptions of projects can be created, even when project clusters are not perfect, (2) accuracy of project clusters improves when clusters are initialized using workstation-wide word distributions gathered from user-provided project names and Google Desktop Search, (3) accuracy of clusters improves further when social network analysis is used to split clusters according to subcommunities of email senders and recipients, and (4) word vectors calculated from workstation-wide data, using Google Desktop Search, can be used to associate calendar meetings and person names with the projects found by ActivityExtractor. In future work we intend to experiment further with jointly clustering the word vectors describing people, meetings, and emails. More generally, we envision many potential uses of these word distribution vectors as the basis for reasoning about people, emails, meetings, files, webpages, and relations among them.

9. ACKNOWLEDGMENTS

Our thanks William Cohen and Vitor Carvalho for providing their MinorThird information extraction and classification software package, and for helpful technical discussions throughout this project. We also thank Darpa for funding this research as part of the PAL project.

10. REFERENCES

[1] [Wu 2003] Wu, F., Huberman, B. A., Adamic, L. A., & Tyler, J. R. (2003). Information flow in social groups.

[2] [Lorrain 1971] Lorrain, F., & White, H. C. (1971). The structural equivalence of individuals in social networks. Journal of MathematicalSociology, 1, 49–80.

[3] [Albert 2002] Albert, R., & Barab´asi, A.-L. (2002). Statistical mechanics of complex networks. Reviews of Modern Physics, 74, 47–97.

[4] [Wasserman 1994] Wasserman, S., & Faust, K. (1994). Social network analysis: Methods and applications. New York: Cambridge University Press.

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

[6] [EmailNet 2004] EmailNet: A System for Automatically Mining Social Networks from Organizational Email Communication. 2004

[7] [Culotta 2004] Aron Culotta, Ron Bekkerman, and Andrew McCallum(2004) Extracting social networks and contact information from email and the Web.

[8] [Ford 1962] L. Ford and D. Fulkerson (1962). Flows in Networks. Princeton University Press.

[9] [Huang, 2004] Y. Huang, D. Govindaraju, T. M. Mitchell, V. R. Carvalho, W. W. Cohen. Inferring Ongoing Activities of Workstation Users by Clustering Email *First Conference on Email and Spam*, Mountain View, CA, July 2004.

[10] [Cohen 2004] William W. Cohen, Vitor R. Carvalho & Tom Mitchell. Learning to Classify Email into "Speech Acts", *EMNLP* 2004

[11] [William 1998] William J. Cook, William H. Cunningham,William R. Pulleyblank, and Alexander Schrijver. Combinatorial Optimization. John Wiley & Sons, 1998.

[12] [McCallum 2004] Andrew McCallum, Andres Corrada-Emmanuel, Xuerui Wang. The Author-Recipient-Topic Model for Topic and Role Discovery in Social Networks: Experiments with Enron and Academic Email. Technical Report UM-CS-2004-096, 2004

[13] [McCallum 2005] Andrew McCallum, Andres Corrada-Emmanuel and Xuerui Wang, Topic and Role Discovery in Social Networks, *IJCAI*, 2005

[14] [Dinic 1970] E. A. Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. Soviet Math. Dokl., 11:1277–1280, 1970.

[15] [Goldberg 1988] A. Goldberg and R. Tarjan. A new approach to the maximum flow problem. Journal of the Association for Computing Machinery, 35(4):921–940, October 1988.

[16] [Rohall 2004] S. Rohall, D. Gruen, P. Moody, M. Wattenberg, M. Stern, B. Kerr, B. Stachel, D. Kushal, R. Armes, and E. Wilcox. Remail: A reinvented email prototype. In Proc. Conf. Human Factors in Computing Systems, 2004.

[17] [Mock 2001] K. Mock. An experimental framework for email categorization and management. In Proc. Int. Conf. Research and Development in Information Retrieval, 2001.

[18] [Kushmerick, 2004] Nicholas Kushmerick, Tessa Lau. Automated Email Activity Management: An Unsupervised Learning Approach, 2004.

[19] [Tyler, 2003] Joshua R. Tyler, Dennis M. Wilkinson, and Bernardo A. Huberman. Email as spectroscopy: Automated discovery of community structure within organizations. Technical report, Hewlett-Packard Labs, 2003.

[20] [Geyer, 2004] Werner Geyer, Juergen Vogel, Li-Te Cheng, Michael Muller. Supporting Activity-centric Collaboration through Peer-to-Peer Shared Objects. GROUP 2003.

[21] [Nigam 2000] Kamal Nigam, Andrew McCallum, Sebastian Thrun and Tom Mitchell.Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning,* 2000.

[22] [Cheyer, 2005] Adam Cheyer, Jack Park, Rich Giuli. IRIS: Integrate. Relate. Infer. Share. ISWC 2005.

[23] [Karger, 2005] David R. Karger., Karun Bakshi, David Huynh, Dennis Quan, and Vineet Sinha,. Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data. CIDR 2005.

[24] [Jetbrains] OMEA, the Integrated Office Environment. http://www.jetbrains.com/omea/

[25] [Surendran, 2005] Arun C. Surendran, John C. Platt, Erin Renshaw. Automatic Discovery of Personal Topics To Organize Email. CEAS 2005

[26] [Kapor, 2005] Mitch Kapor. What's Compelling about Chandler, A Current Perspective. 2005. http://www.osafoundation.org/Chandler_Compelling_Vision.htm