An Email and Meeting Assistant using Graph Walks

Einat Minkov
Language Technologies Inst.
Carnegie Mellon University
einat@cs.cmu.edu

William W. Cohen
Machine Learning Dep.
Carnegie Mellon University
wcohen@cs.cmu.edu

ABSTRACT

We describe a framework for representing email as well as meeting information as a joint graph. In the graph, documents and meeting descriptions are connected via other non-textual objects representing the underlying structure-rich data. This framework integrates content, social networks and a timeline in a structural graph. Extended similarity metrics for objects embedded in the graph can be derived using a lazy graph walk paradigm. In this paper we evaluate this general framework for two meeting and email related tasks. A novel task considered is finding email-addresses of relevant attendees for a given meeting. Another task we define and evaluate is finding the full set of email-address aliases for a person, given the corresponding name string. The experimental results show promise of this approach over other possible methods.

1. INTRODUCTION

Email corpora implicitly represent social network information, textual content and a timeline. Obviously, there is a close relationship between these components of information. For example, persons on a user's contact list may be related by being part of one social "clique", as derived by a simple analysis of header information in a corpus [10, 9]. Or, they can be related via common key words that appear in the relevant correspondece in the email corpus [14]. Such persons' relatedness is also tied to a time dimension. It is therefore desired to combine multiple relevance measures to utilize the multi-faceted information that is included in email for email-prcessing tasks.

We represent email as a graph. The suggested graph scheme naturally models an email corpus in the sense that it forms a direct layout of the information included within the corpus. The graph entities correspond to objects of particular types, including documents and terms, as well as email addresses, persons and dates. The graph edges are directed and correspond to relations like *sent-by*, *sent-on-date* etc. We use this framework to derive a similarity metric between email entities. Our similarity metric is based on a lazy graph walk, and is closely related to the well-known PageRank algorithm [18]. PageRank and its variants (e.g., [8]) are based on a graph walk of infinite length with random resets. In a *lazy* graph walk, there is a fixed probability of halting the walk at each step. In previous work [21], lazy walks over graphs were used for estimating word dependency distributions: in this case, the graph was one constructed especially for this task, and the edges in the graph represented different flavors of word-to-word similarity. Other recent papers have also used walks over graphs for query expansion [22, 6]. In these tasks, the walk propagates similarity to a start node through edges in the graph—incidentally accumulating evidence of similarity over multiple connecting paths.

We view the similarity metric as a *tool for performing search* across this structured dataset, in which related entities that are not directly similar to a query can be reached via multi-step graph walk.

In a previous work [15], this framework has been shown to be very effective for a couple of email processing tasks: namely, email *threading* and *person name disambiguation*. In this paper we extend this framework to represent also *meeting* information. We show that corpora of meeting descriptions can be integrated into the email representing graph. Combining these two information sources may be beneficial in several respects. For example, social analysis can benefit from information about meetings attendees (in addition to email correspondence) and meeting management can be leveraged by linkage to email content.

In this paper, we describe a graph that jointly represents email and meetings. We formulate and evaluate the joint graph for the novel task of *finding relevant meeting attendees*. In this task, we search for relevant meeting attendees given a short text description of the meeting. In particular, we are interested in the attendees' email-address information that can be readily used in coordinating a meeting. A second task that is explored in this paper is finding the set of a person's email-addresses given the person's *name*. This task is closely related to the first task of facilitating meeting management. However, we believe this task to be of interest also as a stand-alone application. Both tasks are phrased as a search query in our framework, where we show that the graph-based approach improves substantially over plausible baselines.

This paper proceeds as follows. In Section 2 we describe the schema for representing email as graph. Section 3 discusses the extension of the graph to include meeting descriptions, where information is represented jointly. Section 4 reviews the formal details of our framework. We then describe the evaluation corpus in Section 5. The evaluation details for the task of finding email-addresses of relevant meeting attendees and for the task of finding a person's email-addresses are given in Section 6 and 7, respectively . We then review related work and conclude.

CEAS 2006 - Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California USA

source type	edge type	target type
file	sent-from	*person*
	sent-from-email	*email-address*
	sent-to	*person*
	sent-to-email	*email-address*
	date-of	*date*
	has-subject-term	*term*
	has-term	*term*
person	sent-from^{-1}	*file*
	sent-to^{-1}	*file*
	alias	*email-address*
	as-term	*term*
email-address	sent-to-email^{-1}	*file*
	sent-from-email^{-1}	*file*
	alias^{-1}	*person*
	email-as-term	*term*
	is-email^{-1}	*term*
term	has-term^{-1}	*file*
	has subject-term^{-1}	*file*
	is-email	*email-address*
	as-term^{-1}	*person*
	email-as-term^{-1}	*email-address*
date	date-of^{-1}	*file*

Table 1: Email graph structure: Node and relation types

2. EMAIL AS A GRAPH

A graph G consists of a set of nodes, and a set of labeled directed edges. Nodes will be denoted by letters like x, y, or z, and we will denote an edge from x to y with label ℓ as $x \xrightarrow{\ell} y$. Every node x has a type, denoted $T(x)$, and we will assume that there is a fixed set of possible node types. We will assume for convenience that there are no edges from a node to itself (this assumption can be easily relaxed.)

We will use these graphs to represent real-world data. Each node represents some real-world entity, and each edge $x \xrightarrow{\ell} y$ asserts that some binary relation $\ell(x, y)$ holds. The entity types used here to represent an email corpus are shown in the leftmost column of Table 1. They include the traditional types in information retrieval systems, namely *file* and *term*. In addition, however, they include the types *person*, *email-address* and *date*. These entities are constructed from a collection of email messages in the obvious way–for example, a receipient of "Einat Minkov <einat@cs.cmu.edu>" indicates the existence of a person node "Einat Minkov" and an email-address node "einat@cs.cmu.edu". (We assume here that person names are unique identifiers.)

The graph edges are directed. We also create an *inverse label* ℓ^{-1} for each edge label (relation) ℓ. Note that this means that the graph will definitely be cyclic.

Table 1 gives the full set of relations used in our email represention scheme. Most relations correspond directly with email structure. In addition, a person node is linked to its constituent token values with an "as-term" edge; Similarly, an email-address node is linked to the constituent tokens of its prefix with an "email-as-term" edge; and a term that is identified as an email-address is linked to an email-address node of the same string value with an "is-email" edge.

3. INCORPORATING MEETINGS

We are now interested in incorporating meeting objects to create a joint graph representing both email and meeting information. Like email, meetings can be represented as a graph. Table 2 gives a possible scheme for meeting representation in terms of entities and the relations between them. In this description we assume that a given meeting includes attendees' information (names and email-addresses), text describing the meeting (e.g.,"Webmaster mtg, 3305 NS") and a date. One can imagine a richer setting where meetings are also linked to files, web URLs, etc.

source type	edge type	target type
meeting	attendee	*person*
	attendee-email	*email-address*
	date-of	*date*
	has-term	*term*
person	attendee^{-1}	*meeting*
	alias	*email-address*
	as-term	*term*
email-address	attendee-email^{-1}	*meeting*
	alias^{-1}	*person*
	is-email^{-1}	*term*
	email-as-term^{-1}	*term*
term	has-term^{-1}	*meeting*
	is-email	*email-address*
	as-term^{-1}	*person*
	email-as-term^{-1}	*email-address*
date	date-of^{-1}	*file*

Table 2: Meetings node and relation types

Evidently, related email and meeting corpora have many entities in common: namely persons and email-addresses, terms and dates. It is therefore straightforward to join the two information sources in a shared graph. In such a graph, a meeting will have a connecting path via term and date nodes to email files, for example. Relevant emails or other potentially included entities, like papers and presentations, can be identified as background material for a meeting's participants in this framework. Similarly, the social network information embedded in emails can be enhanced with meeting information. That is, if meetings in the graph are linked to known attendees, these links may provide additional knowledge about persons' relationships, complementing the social network derived from email files.

Figure 1 shows the graph structure where both email files and meeting files are included. In the figure, we assume that meeting attendees are not known. However, it is easy to integrate additional entities and relation types into the graph representation.

4. GRAPH SIMILARITY

This section describes the formal details of the used framework, including graph edge weighting and the graph walk paradigm.

4.1 Edge weights

Similarity between two nodes is defined by a lazy walk process, where a walk on the graph is controlled by a small set of parameters Θ. To walk away from a node x, one first picks an edge label ℓ; then, given ℓ, one picks a node y such that $x \xrightarrow{\ell} y$. We assume that the probability of picking the label ℓ depends only on the type $T(x)$ of the node x, i.e., that the outgoing probability from node x of following an edge type ℓ is:

$$Pr(\ell \mid x) = Pr(l \mid T_i) \equiv \theta_{\ell,T_i}$$

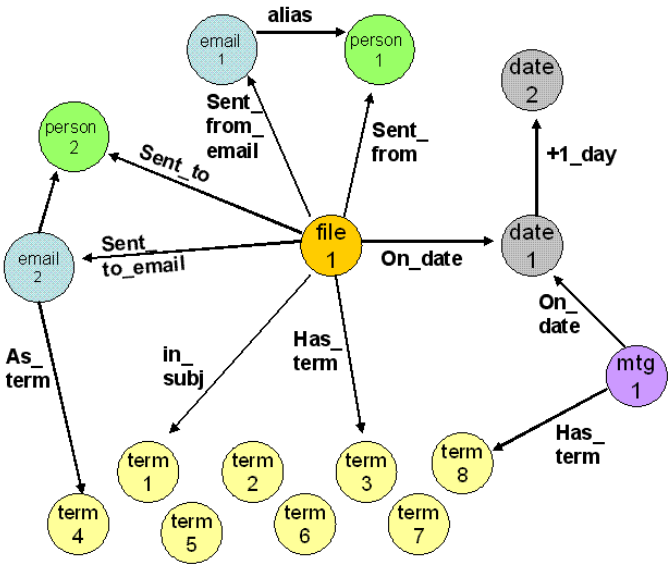


Figure 1: Email and Meetings graph sketch

Let S_{T_i} be the set of possible labels for an edge leaving a node of type T_i. We require that the weights over all outgoing edge types given the source node type form a probability distribution, i.e., that

$$\sum_{\ell \in S_{T_i}} \theta_{\ell,T_i} = 1$$

In this paper, we will assume that once ℓ is picked, y is chosen uniformly from the set of all y such that $x \xrightarrow{\ell} y$. That is, the weight of an edge of type l connecting source node x to node y is:

$$Pr(x \xrightarrow{\ell} y \mid \ell) = \frac{\theta_{\ell,T_i}}{\mid y : x \xrightarrow{\ell} y \mid}$$

This assumption could easily be generalized, however: for instance, for the type $T(x) = \textit{file}$ and $\ell = \textit{has-term}$, weights for terms y such that $x \xrightarrow{\ell} y$ might be distributed according to an appropriate language model [7].

4.2 Graph Walks and Queries

Conceptually, the edge weights above define the probability of moving from a node x to some other node y. At each step in a lazy graph walk, there is also some probability γ of staying at x. Putting these together, and denoting by $\mathbf{M}_{xy}$ the probability of being at node y at time $t+1$ given that one is at x at time t in the walk, we define

$$\mathbf{M}_{xy} = \begin{cases} (1-\gamma) \sum_\ell Pr(x \xrightarrow{\ell} y|\ell) \cdot Pr(\ell|T(x)) & \text{if } x \neq y \\ \gamma & \text{if } x = y \end{cases}$$

If we associate nodes with integers, and make $\mathbf{M}$ a matrix indexed by nodes, then a walk of k steps can then be defined by matrix multiplication: specifically, if V_0 is some initial probability distribution over nodes, then the distribution after a k-step walk is proportional to $V_k = V_0\mathbf{M}^k$. Larger values of γ increase the weight given to shorter paths between x and y. In the experiments reported here, we consider small values of k, and this computation is carried out directly using sparse-matrix multiplication methods.[1] If V_0 gives probability 1 to some node x_0 and probability 0 to all other nodes, then the value given to y in V_k can be interpreted as a similarity measure between x and y.

In our framework, a *query* is an initial distribution V_q over nodes, plus a desired output type T_{out}, and the answer is a list of nodes y of type T_{out}, ranked by their score in the distribution V_k. For instance, for an ordinary *ad hoc* document retrieval query (like "economic impact of recycling tires") would be an appropriate distribution V_q over query terms, with $T_{out} = \textit{file}$. Replacing T_{out} with *person* would find the person most related to the query—e.g., an email contact heavily associated with the retread economics. Replacing V_q with a point distribution over a particular document would find the people most closely associated with the given document.

4.3 Relation to TF-IDF

It is interesting to view this framework in comparison to more traditional IR methods, which can be viewed as a special case. Suppose we restrict ourselves to only two types, terms and files, and allow only *has-term* edges. Now consider an initial query distribution V_q which is uniform over the two terms "the aardvark". A one-step matrix multiplication will result in a distribution V_1, which includes file nodes. The common term "the" will spread its probability mass into small fractions over many file nodes, while the unusual term "aardvark" will spread its weight over only a few files: hence the effect will be similar to use of an IDF weighting scheme.

5. EVALUATION

For evaluation we use a corpus that contains a subset of the second author's email and meeting files. The email files were all drawn from a "meetings" folder, over a time span of about six months. In addition, we use all meeting entries (as maintained in a "Palm" calendar) for the same period. The information available for the meeting files is their accompanying descriptive notes as well as the meeting date. The meeting notes typically include one phrase or sentence – usually mentioning relevant person names, project name, meeting locations etc. The list of attendees per each meeting is not available, and is not included in the constructed graph.

The joint corpus statistics are given in Table 3. The meetings statistics refer to the number of *additional* graph nodes, given that the email information is already represented in the graph.

	corpus		
	instances	**nodes**	**email-addr.**
Email	346	3239	195
Meetings	334	441	-

Table 3: Corpus Details

This corpus is modest in size. We believe that this framework should benefit from larger corpora that may be less

[1]We have also explored an alternative approach based on sampling; this method scales better but introduces some additional variance into the procedure, which is undesirable for experimentation.

sparse in text and have a richer link structure. Nevertheless, despite its size, this experimental corpus is an interesting testbed for the suggested applications.

6. A MEETING ATTENDEES FINDER

6.1 The Task

Having meetings embedded in the graph, one can leverage the information included in both the email and meeting corpora to assist in meeting management. Specifically, we assume that a given meeting is associated with a text description. One can apply a search query starting from a meeting node, looking for relevant email addresses. A returned ranked list of such addresses can be utilized semi-automatically, assisting the user in the task of identifying relevant recipients to include in the meeting invitation or update notifications.

6.2 Dataset

The experimental dataset consists of labeled examples of meetings for which the list of the email addresses of relevant attendees is given (manually annotated by the corpus owner). The number of relevant meeting attendees varies – for some examples that represent personal or small meetings there are only few relevant email-addresses identified, while for larger project meetings there are dozens of relevant email-address nodes. For all examples, all attendees are considered to be equally relevant.

The examples for the time slice of which this corpus was derived are often similar to each other, given that many meetings are periodic. In order to avoid bias towards specific repetitive examples, the constructed dataset includes only 13 examples, manually selected as having distinct attendee lists[2].

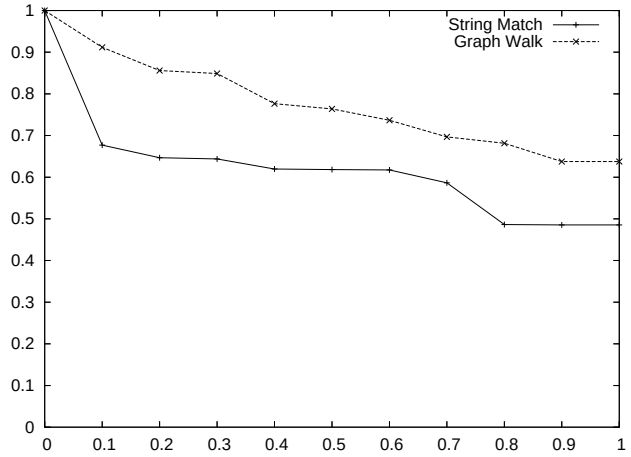
We notice that mapping email-addresses to meetings is not trivial since in many cases, there are multiple email-addresses referring to a single person. Some addresses may be rarely used or obsolete. In the next section describing the experiments we refer to this issue in further detail.

6.3 Experiments

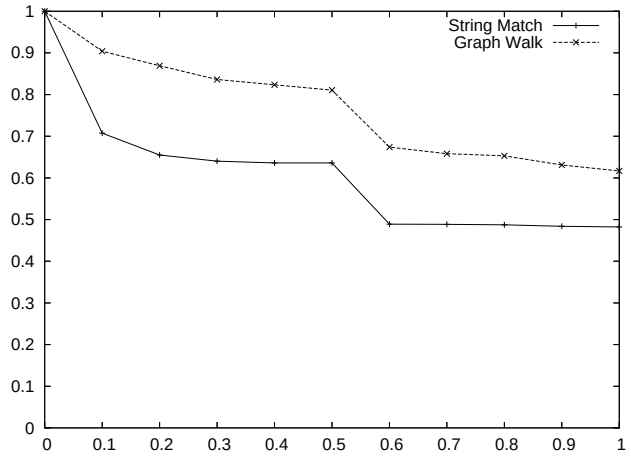
6.3.1 Evaluation Details

All of the methods applied generate a ranked list of email-addresses. Since the number of correct answers varies between examples we use an 11-point precision-recall curve averaged over all examples for evaluation. In case that the ranking results include blocks of items with the same score, a node's rank is counted as the average rank of the "block". Figure 2 gives results for the meeting dataset. The results are given in two forms: the top graph of Figure 1 considers the full list of email-addresses for a relevant person as required correct answers. In contrary, the evaluation in the bottom graph requires *at least* one email-address to be identified per person. For example, suppose a relevant person p has three distinct email-addresses, where the three of them are returned in ranks r_1, r_2 and r_3, having $r_1 < r_2$ and $r_1 < r_3$. In that case, the curve in Figure 2 would consider person p to be ranked at r_1.

[2]We also required that the meetings relate to persons that are likely to appear in the email corpus, as opposed to random visitors.



a. all addresses



b. one address per person

Figure 2: Meeting to attendees email-addresses: Precision-recall curve

6.3.2 Baseline Method

To the best of our knowledge, the suggested task is novel and there is no previous suggested methods in this settings. As a baseline, we use a string matching approach. Since many of the message notes include persons and project names, string matching can utilize the similarity between persons name or public project names and relevant personal or project-related email-addresses. We use the Jaro-Winkler measure [5] to compute string similarity. The similarity score for every email-address is considered as the maximum Jaro-Winkler score of that email-address against any one of the words appearing in a meeting notes. The result of the described procedure is a ranked list of email-addresses, given the meeting notes.

The results of applying the string matching approach are given in Figure 2. As mentioned before, the given meeting notes often include explicit mentions of persons names, which allows string matching to be fairly effective. String matching fails, however, where the meeting text is more general, referring to (formal or informal) project names. In such cases, string matching can not map the given terms to indi-

vidual persons' email-addresses etc.: thus, recall using string matching is necessarily limited.

6.3.3 Graph Walk

We perform a 3-step graph walk. The results of the graph walk are given in Figure 2. The graph walk is clearly preferable in retrieving relevant email addresses given the meeting details.

The graph walk results are considerably better than mere string matching although string similarity is not incorporated into the graph. Unlike string matching, the graph walk can retrieve email-addresses that have no literal ressemblence to a person's name, using co-occurence mappings. In particular, a 3-step walk uses the following paths:

- meeting $\overset{has-term}{\longrightarrow}$ term $\overset{email-as-term^{-1}}{\longrightarrow}$ email-address
- meeting $\overset{has-term}{\longrightarrow}$ term $\overset{as-term^{-1}}{\longrightarrow}$ person $\overset{alias}{\longrightarrow}$ email-address
- meeting $\overset{has-term}{\longrightarrow}$ term $\overset{has-term^{-1}/has-subj-term^{-1}}{\longrightarrow}$ file $\overset{sent-to/from-email}{\longrightarrow}$ email-address
- meeting $\overset{on-date}{\longrightarrow}$ date $\overset{on-date^{-1}}{\longrightarrow}$ file $\overset{sent-to/from-email}{\longrightarrow}$ email-address

The graph walk utilizes the full set of terms in a meeting's description in finding related persons' email-addresses. In addition, a graph walk would give higher weight to frequently-used email-addresses over rarely used ones. Finally, having time information in the graph allows for incorporating a time notion in finding relevant persons. It is straight-forward for example to add edges between date nodes according to time proximity, thus modeling a timeline as required additional graph walk steps. This is not used here, but might be beneficial for richer corpora.

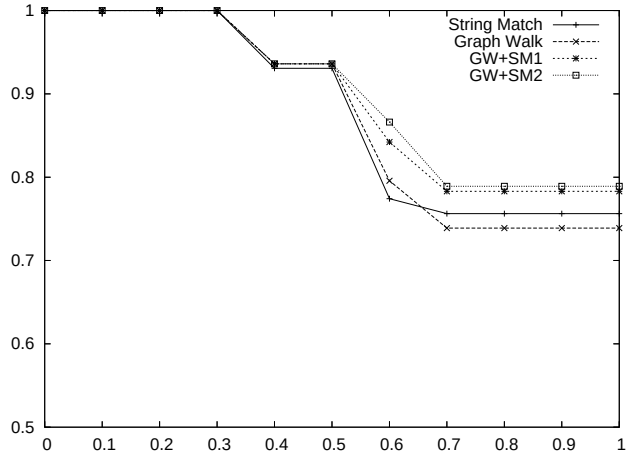
7. EMAIL ADDRESS FINDER

7.1 The Task

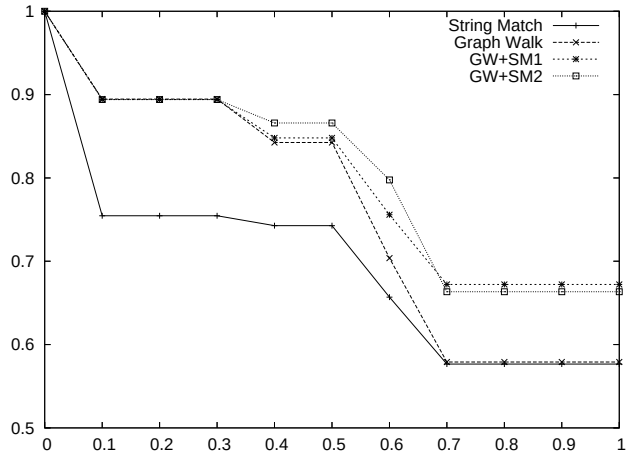
A second related task we consider is automatic assistance in finding a person's email-address. A typical email user oftentimes needs to retrieve email-addresses from his or her address book. In some cases, this is done by searching for an email with the desired information in the header. In this section we evaluate the performance of graph walks for this specific task, comparing it with string matching. We consider two variants of this problem: querying a person's full name to get relevant email-addresses, and querying by the person's first name only. The latter setting may be faster and more convenient for an end user (at least in cases where the person's first name is not ambiguous) and can be used also when a user is not certain about the full name.

7.2 Dataset

We here use the manually labelled list of email-address aliases per person for the given corpus. For the experiment, we constructed a dataset containing 14 examples - all of which refer to individual users (versus mailing lists) that have two to five email-addresses. The relevant person's name was either found in header information or determined based on personal acquaintance. In the experiment, we require the full set of email-addresses to be retrieved given the person's name.



a. full name



b. first name

Figure 3: Person to email-address mapping: Precision-recall curve

7.3 Experiments

7.3.1 Baseline Method

As a baseline, we use here the string matching approach described earlier (section 6.3.2).

The results of applying string matching are given in Figure 3. The top graph of Figure 3 refers to the setting where the full person's name is given as a query. String matching is successful in this case in identifying email-addresses that are similar to either the person's first name or last name. It therefore has relatively high recall. There are, however, email-addresses that are not similar to neither the person's first or last name. Such instances bound the recall of this approach. As can be seen in the bottom graph, string matching performance deteriorates when only the person's first name is given, due to this shortcoming.

7.3.2 Graph Walk

The results of the graph walk are shown in Figure 3. We perform several variants of graph walk.

The curve named 'Graph walk' gives the result of a 3-step

walk. The performance of the graph walk is comparable to string matching when the person's full name is given. The graph walk performance is noticeably better than string matching where only the person's first name is available. The gap stems mainly from the limited recall of the string matching approach, as some of the email-addresses include variants of the person's last name only.

For the reader's convenience, some relevant paths in a 3-step walk are as follows:

- term $\overset{email-as-term^{-1}}{\longrightarrow}$ email-address
- term $\overset{as-term^{-1}}{\longrightarrow}$ person $\overset{alias}{\longrightarrow}$ email-address
- term $\overset{has-term^{-1}/has-subj-term^{-1}}{\longrightarrow}$ file $\overset{sent-to/from-email}{\longrightarrow}$ email-address
- term $\overset{has-term^{-1}/has-subj-term^{-1}}{\longrightarrow}$ file $\overset{sent-to/from}{\longrightarrow}$ person $\overset{alias}{\longrightarrow}$ email-address

The next two variants of graph walk incorporate string matching directly into the graph structure. As above mentioned, the "plain" graph walk is effective in realizing co-occurence information and retrieving highly used email-address nodes. However, rarely used email-addresses may be harder to find using the graph walk approach. Incorporating string matching into the graph links should thus increase graph walk recall.

String similarity is inserted to the graph in the following manner. We add edges between email-address nodes, where the Jaro-Winkler score between the two addresses is above a pre-defined threshold[3]. Specifically, referring to the set of relevant paths described earlier in this section, this edge is added as a "tail" to the previous paths. That is, once the graph walk reaches an email address node, the next step propagates some probability mass to similar email-address nodes over "similar-string" edges.

The results for using the modified graph are given as "GW+SM1" in Figure 3. As expected, the graph walk's overall recall is enhanced due to incorporating string similarity considerations into the graph. "GW+SM2" is another variant, where we assign higher weight to the outgoing edges of type "similar-string" given an email-address node (otherwise weights are distributed uniformly by edge type, as described in section 4.1). The increased weight gives additional improvement to the graph walk precision-recall curve. An automatic optimization of non-uniform graph edge weights has potential for further improving graph walk performance and is a subject for future work.

8. RELATED WORK

As noted above, the similarity measure we use is based on graph-walk techniques which have been adopted by many other researchers for several different tasks. In the information retrieval community, infinite graph walks are prevalent for determining document centrality (e.g., [18, 8, 13]). Another related line of research is of *spreading activation* over semantic or association networks: here the underlying idea is to propagate "activation" from source nodes via weighted links through the network (e.g., [4, 19]). Spreading activation methods are parameterized by user-provided threshold functions for node activation, limits on node distance, preferences over paths, and other constraints. The framework presented here is similar, but operates through unconstrained lazy graph walks, where path preferences can be learned from data [15].

The idea of representing structured data as a graph is widespread in the data mining community, which is mostly concerned with relational or semi-structured data. Recently, the idea of PageRank has been applied to keyword search in structured databases [2]. Analysis of inter-object relationships has been suggested for entity disambiguation for entities in a graph [12], where the graph edges are undirected and edge weights represent confidence in having a connecting path between the entities. It has been suggested to model similarity between objects in relational data in terms of structural-context similarity [11], where the similarity measure corresponds to the expected number of steps required for a random surfer to cross the graph from one object to the other. The latter did not consider edge weights.

As mentioned earlier, not much work has been done that integrates meta-data and text in email. One example examines clustering using multiple types of interactions in co-occurence data [3]. Another recent paper [1] proposes a graph-based approach for email classification. They represent an individual email message as a structured graph representing both content and header, and find a graph profile for each folder; incoming messages are classified into folders using graph matching techniques.

We believe that the tasks formulated in this paper are both novel in the literature. We are not aware of previous works exploring the task of finding a set of relevant meeting attendees (or their email-addresses) in planning or updating a meeting. Previous research focused mainly on automatic *meeting scheduling* (e.g., [20, 16]). Our work facilitates semi-automatic construction of a meeting attendees' list, which is a preliminary step to meeting scheduling. A generative approach has been recently suggested [14] for clustering persons by their inter-similarity assuming a joint model of email recipients and topic. This approach may be adapted to predict relevant persons given text. Our framework, however, is more general. Another recent work [17] uses desktop search to create a bag-of-words representation of email messages, and also of persons and meetings. By this method, cosine similarity between the bag-of-words representation of a meeting and a person can be used to identify relevant meeting attendees. One difference between their approach and ours is that we consider the data structure in evaluating entity relationships; also, we can tune the importance of particular connecting paths in this framework.

Finally, the task of finding a person's set of email-addresses in an email corpus given the person's name is novel as well. This task is related, however, to the task of identifying email aliases in a corpus. Previous works [9, 10] attempted to combine social network information and string similarity measures to identify email aliases. Our approach allows integration of header information and string similarity measures, as used in these works, as well as email content and time in a unified framework.

9. CONCLUSION

We have presented a scheme for representing a corpus of

[3]It may be beneficial to use the exact similarity score on a weighted edge when walking these edges. Here, however, the similarity coding is binary, in one line with the general framework.

email messages as well as meeting entries in a unified graph of typed entities. An extended similarity measure between graph entities is derived via graph walk. We have shown that this scheme provides good performance on two representative meeting and email assisting tasks: constructing a list of relevant meeting attendees, and email-address mapping to person names. The experimental results are promising: for meeting attendee identification, the retrieved email-address ranked list has about 63% precision at full recall comparing with about 49% using string matching; for email-address identification given a person's first name, precision is about 67% at full recall using this framework comparing to 58% using string matching. The performance of graph walks in terms of the precision-recall curve is considerably preferable. These results are especially encouraging taking into account that the experimental corpus is small and therefore relatively sparse. In general, the graphical framework should benefit from increased entity linkage evidence.

In future work, we plan to further explore the scalability of the approach, and also ways of integrating this approach with language-modeling approaches for document representation and document retrieval. Formally, this can be done straightforwardly by appropriately defining $\Pr(d \xrightarrow{\ell} t|\ell)$ for the edge type $\ell = has - term$ to correspond to the probability for t assigned by a language model for the document d, and by defining $\Pr(t \xrightarrow{\ell} d|\ell)$ for the edge type $\ell = has - term^{-1}$ to reflect the probability of the document d given the query term t.

Practically, there are many implementation issues to address. However, a new version of this system (not the one used in the experiments) uses a sampling-based approximation to iterative matrix multiplication. In preliminary timing experiments, the new system can very accurately approximate walks of the sort considered here in around 0.5 seconds, and can approximate 10-step walks on a million-node corpus (from a different domain) in around 10-15 seconds.

Another venue for future work is tuning the graph weights to improve performance. Optimal edge weights should differ depending on the task at hand. Given labeled examples, we are interested in exploring *learning* of edge weight parameters in this framework. Note that a gradient descent approach has already been proposed for similar settings [8]. In addition, re-ranking of the output nodes using graph-walk describing features has proved to be successful in this framework [15]. Given the limited size of the experimental corpus, the re-ranking scheme has not been applied here.

In conclusion, the suggested framework is appropriate for perfoming various search-related tasks in email. Preserving entity type allows one to formulate a broad range of problems as typed search queries—including, in this paper, retrieval of email-addresses related to given meeting or persons names. Other useful tasks in this framework would be, for example, finding terms related to a particular person in a corpus (this can be useful in determining a person's social role), identifying similar email messages given a message as means of providing a user with relevant context etc. This model provides a unified framework for integration of multiple types of information, including social networks, text, timelines as well as other potentially available information sources such as organization charts or other relations of interest.

10. REFERENCES

[1] M. Aery and S. Chakravarthy. emailsift: Email classification based on structure and content. In *ICDM*, 2005.

[2] A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: Authority-based keyword search in databases. In *VLDB*, 2004.

[3] R. Bekkerman, R. El-Yaniv, and A. McCallum. Multi-way distributional clustering via pairwise interactions. In *ICML*, 2005.

[4] H. Berger, M. Dittenbach, and D. Merkl. An adaptive information retrieval system. based on associative networks. In *APCCM*, 2004.

[5] W. W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWEB*, 2003.

[6] K. Collins-Thompson and J. Callan. Query expansion using random walk models. In *CIKM*, 2005.

[7] W. B. Croft and J. Lafferty. *Language Modeling for Information Retrieval*. Springer, 2003.

[8] M. Diligenti, M. Gori, and M. Maggini. Learning web page scores by error back-propagation. In *IJCAI*, 2005.

[9] R. Holzer, B. Malin, and L. Sweeney. Email alias detection using social network analysis. In *LinkKDD*, 2005.

[10] P. Hsiung, A. Moore, D. Neill, and J. Schneider. Alias detection in link data sets. In *Proceedings of the International Conference on Intelligence Analysis*, May 2005.

[11] G. Jeh and J. Widom. Simrank: A measure of structural-context similarity. In *SIGKDD*, 2002.

[12] D. Kalashnikov, S. Mehrotra, and Z. Chen. Exploiting relationship for domain independent data cleaning. In *SIAM*, 2005.

[13] O. Kurland and L. Lee. Pagerank without hyperlinks: Structural re-ranking using links induced by language models. In *SIGIR*, 2005.

[14] A. McCallum, A. Corrada-Emmanuel, and X. Wang. Topic and role discovery in social networks. In *IJCAI*, 2005.

[15] E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual search and name disambiguation in email using graphs. In *SIGIR*, 2006.

[16] T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7), 1994.

[17] T. Mitchell, S. Wang, Y. Huang, and A. Cheyer. Extracting knowledge about users activities from raw workstation contents. In *AAAI*, 2006.

[18] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Technical Report, Computer Science department, Stanford University*, 1998.

[19] G. Salton and C. Buckley. On the use of spreading activation methods in automatic information retrieval. In *SIGIR*, 1988.

[20] S. Sen. Developing an automated distributed meeting scheduler. *IEEE Expert*, 12(4), 1997.

[21] K. Toutanova, C. D. Manning, and A. Y. Ng. Learning random walk models for inducing word dependency distributions. In *ICML*, 2004.

[22] W. Xi, E. A. Fox, W. P. Fan, B. Zhang, Z. Chen, J. Yan, and D. Zhuang. Simfusion: Measuring similarity using unified relationship matrix. In *SIGIR*, 2005.