Extracting social networks and contact information from email and the Web

Aron Culotta
Dept. of Computer Science
University of Mass. Amherst
culotta@cs.umass.edu

Ron Bekkerman
Dept. of Computer Science
University of Mass. Amherst
ronb@cs.umass.edu

Andrew McCallum
Dept. of Computer Science
University of Mass. Amherst
mccallum@cs.umass.edu

Abstract

We present an end-to-end system that extracts a user's social network and 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 conditional random fields—a type of probabilistic model well-suited for such information extraction tasks. By recursively calling itself on new people discovered on the Web, the system builds a social network with multiple degrees of separation from the user. Additionally, a set of expertise-describing keywords are extracted and associated with each person. We outline the collection of statistical and learning components that enable this system, and present experimental results on the real email of two users; we also present some new preliminary results with a method of learning transfer, and discuss the capabilities of the system for address-book population, expert-finding, and social network analysis.

Introduction

It is widely held that, while "Internet search" is an extremely important application, "email" is the number one online activity for most users. Despite this, there are surprisingly few advanced email technologies that take advantage of the large amount of information present in a user's inbox.

A person's business effectiveness is often a direct function of his or her ability to leverage the power and expertise of a widely-cast network of acquaintances. Thus electronic address books are increasingly important personal resources for storing contact information of friends, family, and business associates. These address books contain many detailed fields, including street address, phone numbers, homepage URLs, company name, occupation and free-form notes. Recently, some email software also includes relational fields to indicate a social link between two entries. Unfortunately, the task of manually filling in these fields for each entry is tedious and error-prone. One might consider a system that extracts these fields automatically from email messages; however, this approach is limited to the data present in email.

Interestingly, a number of social networking companies have recently been formed to help connect friends[1] and business associates.[2] These companies aim to assist companies in finding employees, clients, and business partners by exploiting the topology of their social network. However, the networks these companies search are limited to the people who sign up for the service. Other companies[3] extract university and company affiliations from news articles and Web sites to create databases of people searchable by company, job title, and educational history, but do not address social connections between people.

In light of these partial solutions, this paper describes a powerful, statistics- and learning-based information extraction system for mining both email messages and the Web to automatically extract a user's social network, and to obtain expertise and contact information for each person in the network. After extracting people names from email messages, our system works to find each person's Web presence, and then extract contact information from these pages using conditional random fields (CRFs), a probabilistic model that has performed well on similar language processing tasks (McCallum & Li 2003; Kristjansson *et al.* 2004). In addition, the system uses an information-theoretic approach to extract keywords for each person that act as a descriptor of his or her expertise.

The system obtains social links by extracting mentions of people from Web pages and creating a link between the owner of the page and the extracted person. The entire system is called recursively on these newly extracted people, thus building a larger network containing "friends of friends of friends." This larger network contains a significantly wider array of expertise and influence, and represents the contacts that the user could efficiently make by relying on current acquaintances to provide introductions.

The system thus provides capabilities and infrastructure for (a) avoiding tedious form-entry of contact and social-linkage information, (b) finding experts in large companies or communities, (c) automatically recommending additional experts to CC in an email message about a particular topic, (d) analyzing the social network graph to find the individ-

---

[1] http://www.friendster.com
[2] http://www.ryze.com, http://www.linkedin.com
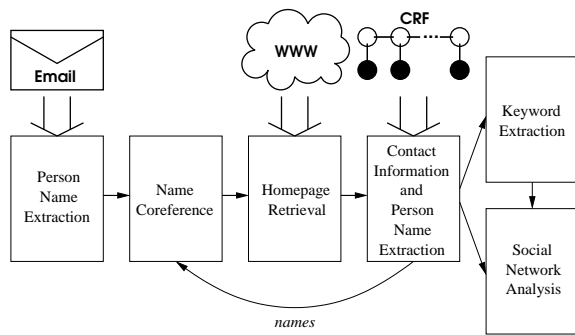[3] http://www.eliyon.com

Figure 1: System overview

uals who are hubs and authorities in a particular sub-field, as well as (e) finding the best path between the user and a desired business or social contact, (f) finding communities with high or low connectivity, (g) clustering people both by their attributes and by graph connectivity.

We demonstrate our system with experimental results on two real-world email datasets from participants in the CALO project (Mark & Perrault 2004). The primary contribution of this paper is to identify the important components of such a system, describe the performance of our first versions of those components, and identify further areas of research opportunity. We also describe and give positive experimental results for a simple method of learning transfer in information extraction—using labeled data from one task to improve performance on another.

In the following sections, we first give a brief overview of the system components, then describe in more detail the theoretical and technical aspects of our methods; we afterwards provide experimental results of the system's performance, and conclude with an outline of future work.

System overview

The system's input is the set of email messages in a user's inbox. The output is an automatically-filled address book of people and their contact information, with keywords describing each person, and links between people defining the user's social network. The six modules of the system are depicted in Figure 1 and briefly outlined below.

1. **Person name extraction.** Names are extracted from the headers of email messages by first locating the header of the email, and then using a set of patterns to find people's names and email addresses.

2. **Name coreference.** A set of string matching rules are used to resolve multiple mentions of the same person. For example, we create rules that will merge people with the names "Joseph Conrad" and "J Conrad".

3. **Homepage retrieval.** We find a person's homepage by creating queries based on the person's name and likely domain. We then submit the query to the Google search engine and filter results based on URL features as well as word distribution similarity metrics between the proposed homepage and the Web page or email messages in which we first found this person.

4. **Contact information and person name extraction.** We employ a probabilistic information extraction model to find contact information and person names in the set of homepages found in the previous step. Extracted people names that are coreferent to people we've already discovered are resolved as in step 2. Links are placed in the social network between a person and the owner of the web page on which the person is discovered. Extracting also from the body of the email is an area of ongoing work.

5. **Expertise keyword extraction.** We create keywords for each person by identifying the terms from each person's homepage that have the highest information gain in separating that person from the others.

6. **Social network analysis.** The resulting social network is clustered into communities by a graph partitioning algorithm that searches for and removes edges that connect two highly-connected components.

The following four sections describe in more detail the techniques employed for finding homepages, extracting contact information, extracting keywords, and clustering the resulting network.

Homepage retrieval

Homepage retrieval consists of locating the Web homepage for a particular person identified either in email or on another Web page. In this system we are particularly interested in finding homepages and not other types of pages because we assume that homepages contain the most up-to-date and well-formatted contact information.

To correctly solve the task, we must find a page that mentions the relevant person, and also determine that it is a homepage. For people with common names, significant care is required to ensure that a page is not about a different person with the same name. For example, there are dozens of homepages for different people named *Tom Mitchell*—among them a few academic researchers in different fields, a jazz guitarist, and a professional photographer. Background knowledge or topical language models may be used in an effort to find the correct one. It can be particularly difficult, however, to determine that none of the candidate pages describe the target person. Determining if a page is a homepage versus other types of pages is a matter of classification – fortunately homepages and their URLs often have distinguishing features for such a classifier.

Our current system finds a person's homepage by the following steps:

1. **Language model construction.** We build a word-count-based language model for the target person, from either the email or Web page text in which we learned of the person's existence.

2. **Query generation and Web search.** We generate an ordered list of Web queries from most specific to most general. The first query uses the most common form of the name (among the names determined to be coreferent) and uses Google's *site:* operator with the most specific domain name appearing in their corresponding email address or Web page domain (for example *"Tom*

Mitchell" site:cs.cmu.edu). If no hits are returned, increasingly general queries are issued; for example, using only the last two components of the domain name (*"Tom Mitchell" site:cmu.edu*), or using alternative name forms, or finally using no domain name restriction. The most general queries can lead to extremely noisy results, especially for people with common names. The queries are issued in order, and the results of the first query to yield non-empty results are passed to the URL-filtering stage.

3. **URL filtering.** To determine if the URL is a homepage, we note that almost every homepage URL contains some version of the person's name. We apply a string kernel distance measure (Lodhi *et al.* 2000) between various forms of the person's name and each field of the URL, and accept homepages within a threshold. These name forms include the full name, first only, last only, and the email login name (if available).

4. **Homepage retrieval.** For each page that passes this filtering, we crawl its internal hyperlink structure to retrieve the user's entire Web home directory. This results in a larger representation of the person's web presence, and frequently provides pages on which contact information and other people's names are located. To prevent overflow, we limit the total number of retrieved pages per person by a reasonably small constant (e.g. 30).

5. **Filtering irrelevant homepages.** If the site is retrieved in response to a query that includes the Internet domain name, then we conclude that the homepage belongs to the person we are looking for, and omit this stage. However, if the query did not include the domain, we compare the word distribution on this homepage site with the language model constructed in step 1. We currently make the comparison with a threshold on cosine similarity between vectors of word counts, which unlike Kullback-Leibler divergence conveniently falls in the range $[0, 1]$ and does not require normalization.

In the Experiments section we compare the results of including or excluding the filtering step, and show that filtering with language models dramatically improves retrieval precision. We also discuss ideas for a more sophisticated approach.

Extracting contact information

To extract contact information from Web pages, we apply a corpus-based, machine learning approach. We first label a training set of documents with 25 fields[4] present in most electronic address books. We then train a conditional random field (CRF) to build a probabilistic model for these fields.

Note that the labels FIRSTNAME, MIDDLENAME and LASTNAME are among the labels predicted by the CRF

---

[4]The 25 fields are FIRSTNAME, MIDDLENAME, LASTNAME, NICKNAME, SUFFIX, TITLE, JOBTITLE, COMPANYNAME, DEPARTMENT, ADDRESSLINE, CITY1, CITY2, STATE, COUNTRY, POSTALCODE, HOMEPHONE, FAX, COMPANYPHONE, DIRECTCOMPANYPHONE, MOBILE, PAGER, VOICEMAIL, URL, EMAIL, INSTANTMESSAGE.

when extracting contact fields. Thus, we can easily group together these tokens into full names to look for mentions of the people in the Web.

Conditional Random Fields

CRFs (Lafferty, McCallum, & Pereira 2001) are undirected graphical models used to calculate the conditional probability of values on designated output nodes given values on designated input nodes. In the special case in which the designated output nodes of the graphical model are linked by edges in a *linear chain*, CRFs make a first-order Markov independence assumption among output nodes, and thus correspond to finite state machines (FSMs). In this case CRFs can be roughly understood as conditionally-trained hidden Markov models, with additional flexibility to effectively take advantage of complex overlapping features.

Let $\mathbf{o} = \langle o_1, o_2, ... o_T \rangle$ be some observed input data sequence, such as a sequence of words in a document (the values on T input nodes of the graphical model). Let $\mathcal{S}$ be a set of FSM states, each of which is associated with a label (such as a label LASTNAME). Let $\mathbf{s} = \langle s_1, s_2, ... s_T \rangle$ be some sequence of states (the values on T output nodes). CRFs define the conditional probability of a state sequence given an input sequence as

$$p_\Lambda(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_\mathbf{o}} \exp \left(\sum_{t=1}^{T} \sum_k \lambda_k f_k(s_{t-1}, s_t, \mathbf{o}, t) \right), \quad (1)$$

where $Z_\mathbf{o}$ is a normalization factor over all state sequences, $f_k(s_{t-1}, s_t, \mathbf{o}, t)$ is an arbitrary feature function over its arguments, and λ_k is a learned weight for each feature function. The normalization factor, $Z_\mathbf{o}$, involves a sum over an exponential number of different possible state sequences, but since these nodes with unknown values are connected in an acyclic graph (a linear chain in this case), it can be efficiently calculated via belief propagation using dynamic programming. Inference to find the most likely state sequence (similar to Viterbi algorithm in this case) is also a simple matter of dynamic programming.

Maximum aposteriori training of these models is efficiently performed by hill-climbing methods such as conjugate gradient, or its improved second-order enhancement, limited-memory BFGS (Sha & Pereira 2003).

Record association

While linear chain CRFs are well-suited to tagging individual contact fields, they do not apply to grouping fields together into one contact record. For example, a document may mention several cities, states, addresses, and companies. When extracting contact fields from free text, it is not always clear which fields should combine together into a single record—for example, which business among several is located in which city.

The system currently addresses this problem by searching for "high concentrations" of contact fields and grouping these into one record. Given the CRF tags in the document, we look for a window of at most N words that contains at least M different contact fields. By looking for M *different* fields, we obtain sections of text that are likely to correspond

to a contact record; and by limiting the window to N words, we reduce the noise introduced by other contact fields in the neighborhood of the contact record. For our experiments, we use $M = 6$ and $N = 15$.

In ongoing work, we are developing a more robust probabilistic model that learns the layout and compatibility of fields within a record (for example, that streets are usually mentioned above city names, Microsoft is usually mentioned with Redmond, Washington, etc).

Learning Transfer

When labeled training data for one task is scarce, it may be desirable to augment it with labeled data or an existing model for some different but related task. By "transferring" knowledge from one task to another, robustness may be increased. This idea is also at the heart of multi-task learning (Caruana 1997), life-long learning (Thrun 1996) and shrinkage (Stein 1955; McCallum *et al.* 1998).

We have currently labeled only 39k words of email and Web data for training a contact extraction system. However, we have obtained 1 million words of newswire text labeled for a traditional named entity extraction (NER) task. This newswire data is related to our contact extraction task in that it includes labels for relevant entities such as people, locations and organizations; it is different in that it is missing labels for the majority of our fields (such as JOBTITLE), in that some of its labels are overly general (such as PERSON instead of FIRSTNAME, MIDDLENAME, LASTNAME), and in that it has some fields not relevant to our task (such as DATE). However, given the overlap, we may hope that information about NER labels would be useful for the contact information extractor.

In this paper we experiment with a straight-forward approach: first, we train a CRF to perform the NER task on a large corpus of newswire articles. Then, we train another CRF for contact information extraction using as additional features the labels predicted by the NER model. The results of this approach are described in the Experiments section. In cases in which the first model has significant uncertainty, preserving that uncertainty via factorial models (Sutton, Rohanimanesh, & McCallum 2004) or finite-state transducer composition (Pereira & Riley 1997) may be helpful.

Extracting keywords

Keywords extracted from a person's homepage play an important role in describing the business area of that person. For example, in a simple application, a user could query his or her social network to find experts in a certain field. More sophisticated applications could identify groups with common interests, key persons within the groups, and other tasks in social network analysis.

We design the keyword extraction module to meet two criteria: (a) keywords should be easily searchable and understandable by humans; (b) keywords should be specific enough to allow the user to distinguish between people with similar interests. We therefore use both unigrams and bigrams, since bigrams carry more semantics than unigrams, and since the sparsity of bigrams helps the system uniquely identify people.

We calculate the Information Gain of a person p and a term t (which can be either a unigram or a bigram) as follows: let $X_p \in \{0, 1\}$ be a binary random variable denoting the event of picking person p from the set of all the people. Let $X_t \in \{0, 1\}$ be a binary random variable denoting the event of term t occurring in a randomly chosen document. The Information Gain of X_p and X_t is then defined as

$$I(X_p, X_t) = \sum_{X_p, X_t \in \{0,1\}} P(X_p, X_t) \log \frac{P(X_p, X_t)}{P(X_p)P(X_t)}. \quad (2)$$

For each person, we first build a list of all the terms (unigrams and bigrams with stopwords removed) that occur on the person's homepage, and then we calculate the Information Gain value for each person-term pair. We sort the lists of terms according to their Information Gain values so that the most "meaningful" terms are ranked at the top of the lists, and then prune the lists to the top few terms.

The Information Gain method is especially appropriate for our task because it takes into consideration not only the event of a term occurring on a homepage but also the event of a term *not* occurring on a homepage, which helps us find terms that are unique to a particular person.

Clustering

After completing the extraction process, we have a social network in which a link between two people indicates that one is mentioned on the other's homepage. More complex models could also indicate additional relationships. Given the potential size of such a network (up to 8,000 people in our experiments), it is useful to cluster the network to discover communities and hubs.

Given that our edges currently have no attributes—not even weights—we then can base our clustering algorithm on the work of Tyler, Wilkinson, & Huberman (2003). This algorithm relies on the notion of *betweenness centrality* (Freeman 1977). Given all shortest-paths between all vertices, the betweenness of an edge is the number of shortest paths that traverse it. The idea is that edges of high betweenness connect people from two distinct communities, while edges of low betweenness connect people within one community.

To efficiently calculate the betweenness of every edge, we rely on a variant of all-pairs-shortest-path, augmented to keep usage counts for edges (Brandes 2001). It proceeds by looping over each vertex v_i and performing Dijkstra's shortest path algorithm (Dijkstra 1959) to calculate the shortest between v_i and all other vertices. The betweenness count for each edge (using only the shortest paths for v_i) is calculated and added to a running total. After each v_i has been considered, the betweenness of each edge is exactly half its running total, since we have considered each pair of endpoints twice. The algorithm runs in time $O(V^2 \lg V + VE)$, where V is the number of vertices and E is the number of edges.

To cluster the graph, we repeatedly remove the edge with highest betweenness. Note that in the case of a tie, we choose an edge randomly. Since different orderings of edge removal result in different clusters, we can iterate this pro-

cess several times to determine which clusters are most consistently discovered.

There are two halting criteria to determine when we should stop removing edges. Tyler, Wilkinson, & Huberman (2003) show that a cluster with 6 nodes is the smallest component that can be split into two clusters. We therefore stop removing edges if all components have fewer than 6 vertices. For components with more than 6 vertices we halt when the highest betweenness of any edge is less than or equal to $N - 1$, where N is the number of vertices in the connected component. The intuition for this is that we do not wish to remove a leaf vertex from the graph, which would result in a cluster of size one. The criterion follows from the observation that the betweenness of any edge connecting a leaf vertex to the rest of the graph of N vertices is $N - 1$.

Note that after removing each edge, we must recalculate the betweenness of all edges in the connected component from which the edge was chosen, since the removal of an edge of high betweenness greatly affects the resulting set of shortest paths. Thus, if we remove K edges, the total running time of the clustering algorithm is $O(K(V^2 \lg V + VE))$. The running time for a network with more than 3,400 nodes is under two hours.

Experiments

We apply our system to the email messages of two participants in the CALO project. Most of the email messages are correspondence between CALO participants on issues related to the CALO project. The data for the first user (**user1**) contains 664 messages. After extracting people names from email headers and resolving coreferent mentions, we obtain 53 individuals, excluding the user. The data for the second user (**user2**) contains 777 messages from which 49 individuals are extracted.

Homepage retrieval results

We report homepage retrieval results on the **user1** dataset, with two recursive iterations of our system. The cosine similarity threshold is set to 0.1. The system finds the web presence of 31 out of 53 email correspondents and retrieves 229 homepages of people listed on the correspondents' homepages, resulting in 260 retrieved homepages within two degrees of the email inbox owner. We manually evaluated all these sites, looking for the following three types of undesirable cases:

1. **People who are not in the user's social network.** We found 16 instances of this type: 7 due to named entity extraction (e.g. *U. S. Healthcare* appearing as a person name), and 4 due to problems of name coreference (e.g. *Raymond Mooney* and *Ray Mooney* not being considered as one person). The remaining 5 errors of this type occur in homepages that mention other people who are not in the owner's social network, such as novelist *Jane Urquhart*. The latter type of error is especially hard to recognize, but is largely addressed in a subsequent stage that separates these people from the user's community by performing clustering in the social network.

2. **Unrelated people with the same name.** We found 25 errors of this type. Many of these pages relate to researchers, sometimes to computer scientists—and in some cases it was even challenging for us to manually determine that the homepage was actually that of an incorrect person. Other errors are caused by the fact that the cosine similarity of some significantly unrelated pages occasionally surpassed the 0.1 threshold.

3. **Relevant page, but not the homepage.** We found 19 people of this type, two of which do not actually maintain a homepage.[5] Most of these mistakes are the result of Web search and URL filtering stages of the homepage retrieval procedure. In some cases a page dedicated to the person was found on the desired domain, however, their actual homepage was on another domain. In other cases the person's username had little in common with the person's real name, so the homepage URL was filtered out.

Thus, not taking into account the first type of anomaly (since it is not a homepage retrieval problem), the precision of finding the relevant person[6] for **user1** dataset is 89.9% and the precision of finding the correct homepage is 82.8%.

Stage 5 of homepage retrieval (filtering irrelevant homepages with a language model) is important for achieving high accuracy. When performing an experiment excluding this stage, the system extracted 621 homepages, 222 of which belong to people who are not in the user's social network (first type of error). Most of these errors (196 of the 222) occurred due to cascading errors from unrelated people found on the first iteration of the algorithm. We also found 92 errors of the second type (namesakes) and 24 errors of the third type. So, the precision of finding homepages in this setting is 47.6%, which represents more than a 35% absolute decrease in relation to the results using stage 5 filtering.

Note, however, that the overall number of correct homepages found without using Stage 5 is 283, in comparison with the previous 202. Thus, while increasing precision, Stage 5 filtering decreases recall. This problem is mostly due to the fact that just a few homepages correctly found at the first iteration fell below the cosine similarity threshold, removing access to a large subgraph of the social network.

We did not obtain qualitatively different results from the **user2** dataset, so we do not report on them due to space restrictions.

Error analysis shows that the most problematic part of homepage retrieval is the cosine similarity thresholding in Stage 5. Now that we have a good understanding of the landscape of this problem, we are investigating more sophisticated probabilistic models for this step, as well as integration with previous steps, and joint inference over multiple people.

Contact information extraction results

To train the CRF, we collected and annotated 2279 files with 25 classes of data fields from various Web and email data,

---

[5] We do not consider these two people as erroneous cases, so we say the total number of errors of this type is 17.

[6] This ignores both the first and third error types.

	Token Acc.	F1	Prec	Rec
CRF	94.24	79.70	86.49	73.90
CRF+NER	94.50	80.76	85.73	76.33

Table 1: Token accuracy and field performance for the original CRF and the CRF trained with the output of a named-entity extractor.

resulting in 26,919 labeled field mentions. About half of the data were isolated address blocks, while the other half were entire Web pages. For testing, we labeled 20 Web pages containing 867 field mentions.

The features consist of the token text, capitalization features, 31 regular expressions over the token text (e.g. CONTAINSHYPHEN, CONTAINSDIGITS, etc.), lexicon memberships, and offsets of these features within a window of the two previous and the two succeeding tokens. We use 25 lexicons, including lists of popular names, cities, companies, job titles, honorifics, and streets. Some of these lexicons were generated automatically by the information extraction system KnowItAll (Etzioni *et al.* 2004).

As noted in the Learning Transfer section, we also train a CRF which uses the output of a NER system as additional features. We refer to this model as CRF+NER. Table 1 displays the per-token accuracy as well as overall field segmentation performance for the two models.

Note that CRF+NER provides a significant boost in recall. Examining results by field, we notice that CRF+NER improves precision considerably for CITY (+31%), DIRECTPHONENUMBER (+22%), and FAXNUMBER (+37%), while actually giving worse performance on COUNTRY (-7%) and MIDDLENAME (-4%). The improvement for CITY makes sense given that NER system labels cities as locations and thus provide a useful feature to CRF+NER. However it is unclear why performance increases for phone number fields.

We perform a paired sign test for token-level accuracy and recall. We find that CRF+NER achieves better token accuracy with significance level $p < 0.13$ and better recall with significance level $p < 1.26e\text{-}6$. This significant improvement in recall is extremely beneficial since the success of the system depends on filling as many fields in the address book as possible, and since the growth of the social network is dependent upon high-recall name extraction.

It may be argued that system precision is still not high enough for effective real-world use. In Kristjansson *et al.* (2004), we propose an interactive extraction system that provides reliable, per-field confidence estimates and incorporates user feedback to enable efficient error correction. Also, we note that segmentation performance given an address block (not an entire Web page) increases F1 by about 10%, which suggests work on an approach that first finds blocks of contact information before labeling the fields.

Keyword extraction results

Applying Information Gain to the problem of extracting keywords for people in the user's social network led to extremely good results. In Table 2, for five well established artificial intelligence researchers who are in **user1**'s social

Researcher	Keywords
William Cohen	logic programming text categorization data integration rule learning
Daphne Koller	bayesian networks relational models probabilistic models hidden variables
Andrew McCallum	information extraction document classification language processing natural language
Deborah McGuinness	semantic web description logics knowledge representation ontologies
Tom Mitchell	machine learning cognitive states learning apprentice artificial intelligence

Table 2: Keywords extracted for some people in **user1**'s social network.

user	$|V|$	$|E|$	max degree	largest component
user1	3377	3402	315	2827
user2	2019	2027	363	2019

Table 3: Statistics of the clusters for two users. $|V|$ is the number of vertices, $|E|$ is the number of edges, *max degree* is the highest degree of any vertex, and *largest component* is the number of vertices in the largest connected component.

network, we list a few highlighting keywords that fell within the top 10 keywords for each person.

Social network analysis

As described previously, we create a social network for each user. See Table 3 for a description of these two networks. Note that in addition to the person links extracted from the Web, we also include links between people who are co-recipients of the same email message.

Figure 2 shows log-log plots of vertex degree k versus the number of vertices with degree k, denoted $n(k)$. We note that both distributions approximately follow a Zipfian distribution with coefficients 4.0 and 5.6, indicating the scale-free properties of the network.

We perform our clustering algorithm to partition the graph into communities. This is practical from a user interface perspective since it allows users to more easily navigate their address books. It is also useful for finding well-connected individuals, which is generally an indicator of the person's standing in the community. Figure 3 displays the clusters discovered for **user2**'s network[7].

For **user1**, the initial graph has 3 large connected components, which we cluster into 116 connected components,

---

[7]To display the graph, we use Zoomgraph, available at http://www.hpl.hp.com/research/idl/projects/graphs/
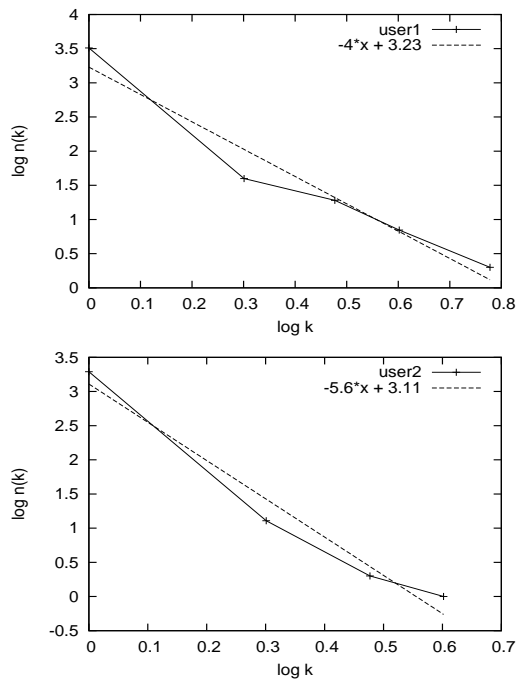
Figure 2: Log-log degree distribution for the social networks extracted for **user1** and **user2**, where k is the degree and $n(k)$ is the number of nodes with degree k. Note that these distributions approximately follow a Zipf distribution.

Figure 3: Clustered social network for **user2**.

with an average cluster size of 29. For **user2**, we cluster the original graph (initially one connected component) into 39 clusters, with an average size of 50.

While evaluating community discovery is an extremely subjective task, we do find intuitively pleasing hubs (e.g. Michael Jordan of UC Berkeley at the center of a machine learning community), and cluster separation between scientific researchers and celebrities. Many clusters consist of researchers at the same university or in the same field (e.g. we found clusters of researchers in astrophysics, information retrieval, and machine learning, as well as University of Massachusetts faculty). It is noteworthy that many of the hubs of these communities are *not* present in the user's email. This provides an example of helping people locate well-connected contacts in communities of interest. Since the system can maintain the original graph structure prior to clustering, the user can recover the shortest path of introductions to make this valuable connection.

Community discovery is a step towards relation finding because clusters often coincide with their relationship to the user. For example, clusters of people working at the same institution as the user can be labeled "people I work with." Automatically generating these labels is a subject of ongoing work.

Related Work

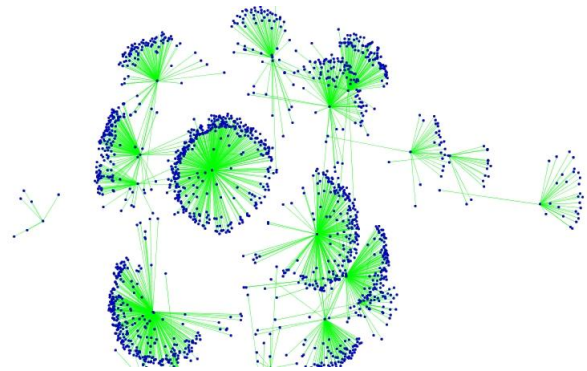While there has been much work in information extraction and social network analysis independently, we believe this is the first paper to propose an end-to-end system that integrates them both, employing the Web to find information about people in a user's email, and extracting both the user's social network and the contact information for people in this network.

Early work on homepage retrieval in the *Ahoy* system (Shakes, Langheinrich, & Etzioni 1997) primarily applies a useful collection of heuristic techniques. Xi *et al.* (2002) give a machine learning approach to homepage retrieval using decision trees and logistic regression, however with moderately low results. Upstill, Craswell, & Hawking (2003) describe an information retrieval technique which augments content-based retrieval with link analysis (e.g. PageRank) and URL features. Since Google incorporates this link analysis into its search, we are effectively employing a PageRank-based filter in our homepage finder.

In related work on contact record extraction, Borkar, Deshmukh, & Sarawagi (2000) obtain high accuracy using an HMM on a significantly simpler and more limited set of fields (HouseNumber, PO Box, Road, City, State, ZIP), which usually appear in very regular form.

Similar social network research has been conducted on Usenet data (Smith 1999), in which the goal is to characterize a dynamic online community as well as determine the "authority" of an individual based on posting patterns. Also, van Alstyne & Zhang (2003) propose a system to analyze the social network of an email graph; however, their approach assumes access to all email messages, not just those of a single user. While this may be practical for a large company, it is infeasible for the ordinary user.

There are many clustering algorithms, including spectral clustering (Ng, Jordan, & Weiss 2001), probabilistic relational models (Taskar, Segal, & Koller 2001), graph partitioning (Ding *et al.* 2001), and probabilistic latent semantic indexing (Cohn & Hofmann 2001). The betweenness clustering approach we use here can be viewed as a type graph partitioning algorithm.

There has also been considerable work in analyzing co-authorship graphs (Newman 2001), which is similar to work in this paper because many relations we find are created by extracting co-author names from the researcher's publications page.

Conclusions

Email is the primary way that people access their widespread social networks. This paper has presented an end-to-end system that automatically integrates both email data and Web content to help users maintain large contact databases, leverage their social network, perform expert finding, and make new relevant connections. The information gathered by the system could also be used as aids to other email functionality, such as automatic foldering and spam detection. By describing the components of such a system and providing early experimental results, we hope to stimulate further research in this field.

Acknowledgments

References

Borkar, V. R.; Deshmukh, K.; and Sarawagi, S. 2000. Automatically extracting structure from free text addresses. In *Bulletin of the IEEE Computer Society Technical committe on Data Engineering*. IEEE.

Brandes, U. 2001. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology* 25(2):163–177.

Caruana, R. 1997. Multitask learning. *Machine Learning Journal* 28(1).

Cohn, D., and Hofmann, T. 2001. The missing link - a probabilistic model of document content and hypertext connectivity. In *Neural Information Processing Systems 13*.

Dijkstra, E. W. 1959. A note on two problems in connection with graphs. *Numerische Math* 1:269–271.

Ding, C. H. Q.; He, X.; Zha, H.; Gu, M.; and Simon, H. D. 2001. A min-max cut algorithm for graph partitioning and data clustering. In *Proceedings of ICDM 2001*, 107–114.

Etzioni, O.; Cafarella, M.; Downey, D.; Popescu, A.-M.; Shaked, T.; Soderland, S.; Weld, D. S.; and Yates, A. 2004. Methods for domain-indepedent information extraction from the web: An experimental comparison. In *Proceedings of AAAI*.

Freeman, L. 1977. A set of measures of centrality based on betweenness. *Sociometry* 40:35–41.

Kristjansson, T.; Culotta, A.; Viola, P.; and McCallum, A. 2004. Interactive information extraction with constrained conditional random fields. In *AAAI*.

Lafferty, J.; McCallum, A.; and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, 282–289. Morgan Kaufmann, San Francisco, CA.

Lodhi, H.; Shawe-Taylor, J.; Cristianini, N.; and Watkins, C. 2000. Text classification using string kernels. In *Advances in Neural Information Processing Systems (NIPS)*, 563–569.

Mark, W., and Perrault, R. 2004. CALO: a cognitive agent that learns and organizes. https://www.calo.sri.com.

McCallum, A., and Li, W. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In Hearst, M., and Ostendorf, M., eds., *HLT-NAACL*. Edmonton, Alberta, Canada: Association for Computational Linguistics.

McCallum, A. K.; Rosenfeld, R.; Mitchell, T. M.; and Ng, A. Y. 1998. Improving text classification by shrinkage in a hierarchy of classes. In Shavlik, J. W., ed., *Proceedings of ICML-98, 15th International Conference on Machine Learning*, 359–367. Madison, US: Morgan Kaufmann Publishers, San Francisco, US.

Newman, M. 2001. Who is the best connected scientist? a study of scientific coauthorship networks. *Phys. Rev.* 64.

Ng, A.; Jordan, M.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *Neural Information Processing Systems 14*.

Pereira, F., and Riley, M. 1997. Speech recognition by composition of weighted finite automata. In Roche, E., and Schabes, Y., eds., *Finite-State language processing*. MIT Press. 431–453.

Sha, F., and Pereira, F. 2003. Shallow parsing with conditional random fields. In Hearst, M., and Ostendorf, M., eds., *HLT-NAACL: Main Proceedings*, 213–220. Edmonton, Alberta, Canada: Association for Computational Linguistics.

Shakes, J.; Langheinrich, M.; and Etzioni, O. 1997. Dynamic reference sifting: A case study in the homepage domain. In *Proceedings of the 6th World Wide Web Conference*.

Smith, M. 1999. Invisible crowds in cyberspace: Measuring and mapping the social structure of usenet. In Smith, M., and Kollock, P., eds., *Communities in Cyberspace*. Routledge Press.

Stein, C. 1955. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability 1*, 197–206. University of California Press.

Sutton, C.; Rohanimanesh, K.; and McCallum, A. 2004. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of ICML*.

Taskar, B.; Segal, E.; and Koller, D. 2001. Probabilistic classification and clustering in relational data. In Nebel, B., ed., *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence*, 870–878.

Thrun, S. 1996. *Explanation-Based Neural Network Learning: A Lifelong Learning Approach*. Boston, MA: Kluwer Academic Publishers.

Tyler, J. R.; Wilkinson, D. M.; and Huberman, B. A. 2003. Email as spectroscopy: Automated discovery of community structure within organizations. Technical report, Hewlett-Packard Labs.

Upstill, T.; Craswell, N.; and Hawking, D. 2003. Query-independent evidence in home page finding. In *ACM Transactions On Information Systems*.

van Alstyne, M., and Zhang, J. 2003. Emailnet: A system for automatically mining social networks from organizational email communication. In *NAACSOS2003*.

Xi, W.; Fox, E. A.; Shu, J.; and Tan, R. 2002. Machine learning approach for homepage finding task. In *Proceeding of the 9th International Symposium on String Processing and Information Retrieval*, 145–159.