Mining Community Structure of Named Entities from Free Text

Xin Li
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan (M/C 152)
Chicago, IL 60607-7053, USA
1 (312) 355-1318
xli3@cs.uic.edu

Bing Liu
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan (M/C 152)
Chicago, IL 60607-7053, USA
1 (312) 355-1318
liub@cs.uic.edu

ABSTRACT
Although community discovery has been studied extensively in the Web environment, limited research has been done in the case of free text. Co-occurrence of words and entities in sentences and documents usually implies connections among them. In this paper, we investigate the co-occurrences of named entities in text, and mine communities among these entities. We show that identifying communities from free text can be transformed into a graph clustering problem. A hierarchical clustering algorithm is then proposed. Our experiment shows that the algorithm is effective to discover named entity communities from text documents.

Categories and Subject Descriptors
H.3 [**Information storage and retrieval**]: General
I.5.3 [**Clustering**]: Algorithms.

General Terms
Algorithms, Experimentation.

Keywords
community, graph clustering, named entities.

1. INTRODUCTION
Knowledge discovery in social networks has attracted much attention due to its successful application in Web search engines. PageRank [1] and HITS [5] are two well-known Web page ranking algorithms. They regard a collection of Web pages as a social network, and evaluate individual page importance by analyzing the Web link structure.

In addition, HITS discovered that there exist multiple Web communities among relevant Web pages when the query term has multiple meanings. A Web *community* can be defined as a group of Web pages that are more closely linked to their group peers than those outside of the group. It usually consists of Web pages with the same theme. A typical example is that query "Jaguar" returns three major Web communities, respectively on the video game, the Football team, and the automobile model.

CIKM'05, October 31-November 5, 2005, Bremen, Germany.
ACM 1-59593-140-6/05/0010.

We believe that the concept of community also exists in free text. Different from Web pages, text documents do not contain any explicit linkage. However, there are many named entity terms in text documents. We believe that there are relationships among those named entities co-occurring in the same context, which can be considered as an implicit link between them.

To identify named entity communities from text, we first convert text documents into a named entity graph by mapping each named entity to a vertex, and each sentence containing named entities to an edge. Then we use a bottom up (agglomerative) clustering algorithm to obtain the final communities.

2. RELATED WORK
The work on community structure discovery on the Web first appeared in the HITS algorithm [5]. [3] approached the Web community issue in a Web graph from a local perspective. The authors introduced the concept of "curvature" for each vertex v to measure how well connected v's neighborhood is. Moreover, the authors have made an observation that community expands predominantly by triangles sharing a common side.

The community concept has also been studied in other cases. [2] applied it in Word Sense Disambiguation problem. "Community" in their work consists of words having similar senses.

Another related research focused on clustering relations of named entities in text corpora [4].

3. THE PROPOSED ALGORITHM
There are two major tasks in our work. The first one is to acquire the weighted graph consisting of named entities. Thereafter, we cluster the named entity graph into communities.

3.1 Named Entity Graph Generation
First, we need a collection of documents regarding a certain named entity. We issue major company names to the Financial Times (FT) corpus as queries. If the term frequency in an article is larger than 1, we regard this article as relevant. The number of retrieved article ranges from 300-500. Second, we extract all the sentences containing more than one entity's name from the document collection. Finally, we map the selected sentences into a weighted undirected graph. The vertices in this graph are entity names, and the edges are co-occurrences of names.

Table 1. The Discovered Named Entity Communities

Entities	Cluster Entities	Summary Terms	Remarks
Sony	sony, cbs records, cbs, mca, matsushita, columbia pictures, ge.	1988, company, year, acquisition, chairman, music, product.	Acquisition events
	motorola, sony, apple computer.	product, media, general, magic, technology	Company cooperation
	sony, warner bros., time warner, paramount.	producer, contract, movie, company, year,	Peer entertainment companies
	toshiba, sony, fujitsu, jvc, nec.	japan, company, electronics, usa, phone, industry.	Peer Japanese companies
IBM	ibm, microsoft, nec, toshiba, fujitsu, hitachi, intel, groupe bull, motorola, novell.	computer, market, pc, company, software, chip, manufacturer.	PC makers
	ibm, sun, hp, mips.	workstation, market, risc, technology, standard.	Workstation makers
Citi	citibank, bank of america, reserve bank of india	bank, report, scandal, committee, guideline.	Scandal rumor
	citicorp, o&y, citibank, chase manhattan, olympia & york, SEC, merrill lynch, morgan stanley, bankers trust.	bank, creditor, subsidiary, loan, president, property.	Peer financial companies

3.2 Clustering Graph into Communities

We propose a hierarchical clustering algorithm to generate communities from the named entity graph. Let us introduce some definitions below.

Edge weight: We use the *mutual information* between vertices *a* and *b, I (a,b),* as the edge weight. The following is the equation for calculating *I (a,b)*. *p(a, b)* is the co-occurrence probability of (*a*, *b*). *f(v)*, $v \in V$, is the incidence probability of vertex *v*.

$$I(a,b) = p(a,b) \times \log_2 \frac{p(a,b)}{f(a)f(b)}$$

Triangle: In a graph $G = (V, E)$, if $a \in V$, $b \in V$, $c \in V$, and edges $(a, b) \in E$, $(a, c) \in E$, and $(b, c) \in E$, we say vertices *a, b, c* form a triangle. The *cohesion* of a triangle is measured by the weight of its weakest edge.

Similarity between Triangles: If two triangles share an edge, we assign the sum of the cohesion value of two triangles as their similarity. If two triangles do not share any edge, we assign their similarity to 0. Two triangles are *fully linked* if merging the two triangles by the common edge can generate a complete graph of four vertices.

There are two assumptions for our algorithm:

1. Each community has a *core*, which is composed by strongly linked triangles. The community propagates mainly through triangle pairs sharing common edges [4].
2. Each triangle belongs to only one community, but each vertex and edge could belong to more than one community.

The procedures of our clustering algorithm are as follows:

1. Triangle Extraction. We extract all triangles from the graph adjacency list.
2. Similarity Calculation. From the triangles extracted, we calculate the similarity for all the triangle pairs. Sort the triangle pairs according to their similarity in descending order.
3. Pre-clustering. We extract all fully linked triangle pairs. If any triangle pairs have a common triangle, we merge them together into a cluster. All these clusters formed in this step are labeled as *cores*.
4. Triangle Clustering. We traverse the sorted triangle pairs. For a triangle pair *A* and *B*, we try to merge the triangle pairs together. If both *A* and *B* were already pre-clustered in the same core, we do nothing. If only one of them, say, *A* was pre-clustered to a core, we put *B* into the same core. If both of them were not pre-clustered, we merge them together, but the merged cluster is not labeled as core. If both *A* and *B* were already pre-clustered into different cores, we do nothing. The final results of this step are clusters of triangles.

Clearly, our algorithm uses automatically formed cores as seeds and does not need the number of clusters to be specified by user.

3.3 Experiment Results

Table 1 shows our experiment results. Column 1 gives the entity name. Column 2 lists the community entities in descending order of their importance, which was measured by the mutual information sum between the entity and its community peers.

To summarize a community theme, we extracted nouns from co-occurrence sentences, and listed the top nouns in column 3. We also manually added remarks on the communities in column 4.

4. CONCLUSION

This paper studied the problem of mining communities from free text. By exploiting the named entity co-occurrence, we mapped free text document into a named entity graph. Moreover, we proposed an effective hierarchical clustering algorithm. Our experimental results show that our clustering algorithm can effectively discover interesting communities.

5. REFERENCES

[1] Brin, S. and Page, L. The anatomy of a large-scale hypertext Web search engine. In *Proceedings of the seventh international conference on World Wide Web 7* (Brisbane, Australia, 1998), 107-117.

[2] Dorow, B. and Widdows, D. Discovering corpus-specific word senses. In *EACL*, (Budapest, Hungary, 2003), 79-82.

[3] Eckmann, J., and Moses, E. Curvature of co-links uncovers hidden thematic layers in the World Wide Web. In *Proceedings of the National Academy of Sciences,* (2002) 99:5825-5829.

[4] Hasegawa, T., and Sekine, S., and Grishman, R. Discovering Relations among Named Entities from Large Corpora. In *ACL* (2004), 415-422.

[5] Kleinberg, J. Authoritative sources in a hyperlinked environment. In *Proceedings of ACM-SIAM symposium on discrete algorithms*, (1998), 668 - 677.