Knowledge Management: A Text Mining Approach

Ronen Feldman (ronen@instinct-soft.com),* Moshe Fresko,*
Haym Hirsh, Yonatan Aumann,* Orly Liphstat,***
Yonatan Schler,* Martin Rajman,***

*****Department of Mathematics and Computer Science, Bar-Ilan University, Ramat-Gan, ISRAEL
and Instinct Software Ltd., 11 Ben Gurion St., Givat Shmuel, ISRAEL 51905
**Department of Computer Science, Rutgers University, New Brunswick, NJ 08855
***LIA, EPFL, Lausanne, Switzerland

Abstract

Knowledge Discovery in Databases (KDD), also known as *data mining*, focuses on the computerized exploration of large amounts of data and on the discovery of interesting patterns within them. While most work on KDD has been concerned with structured databases, there has been little work on handling the huge amount of information that is available only in unstructured textual form. Given a collection of text documents, most approaches to *text mining* perform knowledge-discovery operations on labels associated with each document. At one extreme, these labels are keywords that represent the results of non-trivial keyword-labeling processes, and, at the other extreme, these labels are nothing more than a list of the words within the documents of interest. This paper presents an intermediate approach, one that we call *text mining at the term level*, in which knowledge discovery takes place on a more focused collection of words and phrases that are extracted from and label each document. These terms plus additional higher-level entities are then organized in a hierarchical taxonomy and are used in the knowledge discovery process. This paper describes Document Explorer, our tool that implements text mining at the term level. It consists of a document retrieval module, which converts retrieved documents from their native formats into documents represented using the SGML mark-up language used by Document Explorer; a two-stage term-extraction approach, in which terms are first proposed in a term-generation stage, and from which a smaller set are then selected in a term-filtering stage in light of their frequencies of occurrence elsewhere in the collection; our taxonomy-creation tool by which the user can help specify higher-level entities that inform the knowledge-discovery process; and our knowledge-discovery tools for the resulting term-labeled documents. Finally, we evaluate our approach on a collection of patent records as well as Reuters newswire stories. Our results confirm that Text Mining serves as a powerful technique to manage knowledge encapsulated in large document collections.

Keywords: Text Mining, Taxonomy Construction, Term Extraction

1 Introduction

Traditional databases store information in the form of structured records and provide methods for querying them to obtain all records whose content satisfies the user's query. More recently however, researchers in *Knowledge Discovery in Databases* (KDD) have provided a new family of tools for accessing information in databases (e.g. Anand and Khan, 1993; Brachman et al, 1993; Frawley et al, 1991; Klösgen, 1992). The goal of such work, often called *data mining*, has been defined as "the nontrivial extraction of implicit, previously unknown, and potentially useful information from given data" (Piatetsky-Shapiro and Frawley, 1991). Work in this area includes applying machine-learning and statistical-analysis techniques towards the automatic discovery of patterns in databases, as well as providing user-guided environments for exploration of data.

Most efforts in KDD have focused on data mining from structured databases, despite the tremendous amount of online information that appears only in collections of

Proc. of the 2nd Int. Conf. on Practical Aspects of Knowledge Management (PAKM98)
Basel, Switzerland, 29-30 Oct. 1998, (U. Reimer, ed.)

http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-13/

unstructured text. This paper focuses on the problem of *text mining*, performing knowledge discovery from collections of unstructured text. One common technique (Feldman and Dagan, 1995; Feldman and Hirsh, 1996; Feldman et al., 1997) has been to assume that associated with each document is a set of labels and to perform knowledge-discovery operations on the labels of each document. The most common version of this approach has been to assume that labels correspond to keywords, each of which represents that a given document is about the topic associated with that keyword. However, to be effective, this requires either: manual labeling of documents, which is infeasible for large collections; hand-coded rules for recognizing when a label applies to a document, which is difficult for a human to specify accurately and must be repeated anew for every new keyword; or automated approaches that learn from labeled documents rules for labeling future documents, for which the state of the art can guarantee only limited accuracy and which also must be repeated anew for every new keyword. A second approach (Lent et al., 1997) has been to assume that a document is labeled with each of the words that occurs within it. However, as was shown by Rajman and Besançon (1997) and is further supported by the results presented here, the results of the mining process are often rediscoveries of compound nouns (such as that "Wall" and "Street" or that "Ronald" and "Reagan" often co-occur) or of patterns that are at too low a level (such as that "shares" and "securities" co-occur). Other approaches to text mining that perform deep knowledge discovery include (Hahn and Schnattinger, 1997).

In this paper we instead present a middle ground, in which we perform *term extraction* on each document to find word sequences that are likely to have meaning in the domain, and then perform mining on the extracted terms labeling each document. (A fragment of a document with extracted terms underlined is given in Figure 1.) Unlike word-based approaches, the extracted terms are fewer in number and tend to represent more meaningful concepts in the domain of the document. Unlike keyword approaches, our term-extraction method, eliminates much of the difficulties in labeling documents when faced with a new collection or new keywords. As will be described, we exploit the fact that term extraction does not occur in isolation, but rather as part of the mining process, and we therefore exploit the known target of term extraction – for text mining from a *collection* of documents – in the extraction process itself.

Profits at Canada's six big banks topped C$6 billion ($4.4 billion) in 1996, smashing last year's C$5.2 billion ($3.8 billion) record as Canadian Imperial Bank of Commerce and National Bank of Canada wrapped up the earnings season Thursday. The six banks each reported a double-digit jump in net income for a combined profit of C$6.26 billion ($4.6 billion) in fiscal 1996 ended Oct. 31.

But a third straight year of record profits came amid growing public anger over perceived high service charges and credit card rates, and tight lending policies.

Bank officials defended the group's performance, saying that millions of Canadians owned bank shares through mutual funds and pension plans.

Figure 1. Example of the output of the term extraction module. Terms chosen to label the document are underlined.

This paper describes Document Explorer, a system that embodies this approach to *text mining at the term level*. The overall structure of Document Explorer is shown in Figure 2. The first step is to convert documents (either internal documents or external documents fetched) into an SGML format understood by Document Explorer. The resulting documents are then processed to provide additional linguistic information about the contents of each document – such as through part-of-speech tagging. Documents are next labeled with terms extracted directly from the documents, based on syntactic analysis of the documents as well as on their patterns of occurrence in the overall collection. The terms and additional higher-level entities are then placed in a taxonomy through interaction with the user as well as via information provided when documents are initially converted into Document Explorer's SGML format. Finally, KDD operations are performed on the term-labeled documents.

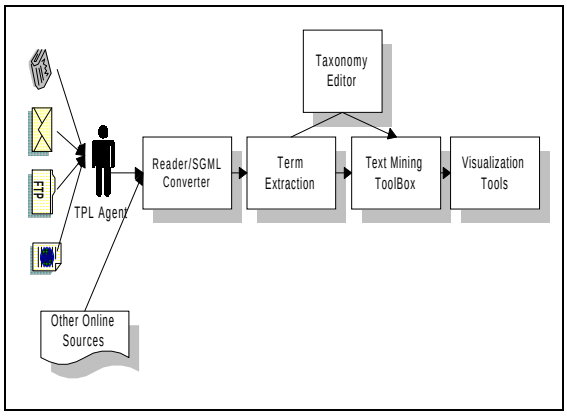

Figure 2. Document Explorer architecture.

Examples of document collections suitable for text mining are documents on the company's Intranet, patent

collections, newswire streams, results returned from a search engine, technical manuals, bug reports, and customer surveys.

In the remainder of this paper we describe Document Explorer's various components. This includes the expected SGML format of the documents to be analyzed, the linguistic preprocessing steps, Document Explorer's two-stage term extraction process, its tool for creating a taxonomic hierarchy for the extracted terms, and, finally, a sample of it's suite of term-based knowledge-discovery tools. We give examples of mining results on a collection of patent records as well as Reuters newswire stories.

2 Document Input Format

Document Explorer needs to be able to easily ascertain a number of features of each document it analyzes. First, it should know the title of each document, if for no other reason than to have a brief description of each document in case it gets displayed in any list of documents returned by a text mining operation. It also needs to know which parts of an overall document should be subject to analysis – for example, in the context of patent records this might be just the text in the abstract and claims portions of the patent. Finally, if a document includes keywords, they must be identified to Document Explorer. Similarly, for those documents that have a well-defined date stamp, the date should be identified as well.

Rather than requiring documents to be in a rigid format specifying all these things, Document Explorer allows its documents to be in a more arbitrary format, using SGML mark-ups, but with an auxiliary file defining which SGML pieces correspond to which components of a document. For example, Figure 3 gives a sample of one document from the patent-records domain. The process by which it was created from the original on-line record was rather simple – indeed, we have defined a simple translation language called TPL that we use to perform such tasks, but it could equally well have been performed through arbitrary means, such as Perl – the reason for the creation of TPL is that we have also added to it the functionality of going to retrieve desired documents over the internet via http and ftp. (For the remainder of this paper we assume – as Document Explorer does – that documents are provided in a suitable SGML-labeled fashion.)

```
<DOCUMENT><ID>1</ID>
<CODE>5694615</CODE>
<SUBJECT>Storage system
having storage units
interconnected to form
multiple loops to provide
simultaneous access from
multiple hosts</SUBJECT>
<INVENTORS>Thapar;
Manu</INVENTORS>
<ADDRESS>Fremont,CA</ADDRESS>
<ASSIGNEES>Hewlett Packard
Company</ASSIGNEES>
<ADDRESS2>Palo Alto,
CA</ADDRESS2>
<ISSUED>2/12/1997</ISSUED>
<FILED>26/7/1995</FILED>
<AGENTS>Short; Brian
R.</AGENTS>
<ABSTRACT>The present
invention is an apparatus and
method for using the dual
port feature of Fibre Channel
to allow multiple computer
hosts to simultaneously
access a cluster of memory
units that are Fibre Channel
arbitrated. Typical multiple
host access schemes require
an expensive Fibre Channel
switch and do not allow
simultaneous accessing. The
dual port feature of Fibre
Channel devices provides for
fault tolerance and
redundancy, but can be used
for the present invention.

</ABSTRACT>
</DOCUMENT>
```

Figure 3. An example patent-record document.

Given such an SGML-labeled document, it is necessary to inform Document Explorer what the various SGML-labeled components designate. This is done in a "tags" file that must be specified for each collection, which explains how the various SGML-designated components should be interpreted. For example, Figure 4 gives an example of the tags file for the patents domain. It is interpreted as saying that each document in a given file (in general Document Explorer assumes all documents are placed in a single rile) begins with <DOCUMENT> and ends with </DOCUMENT>, that the title of each document is the sequence of characters between the <SUBJECT> and </SUBJECT> labels, that the textual body of the document should be taken as the union of all text found between <CLAIMS> and </CLAIM> labels and <ABSTRACT> and </ABSTRACT> labels, and that the sequence of characters between the <INVENTORS> and </INVENTORS> labels, the <ASSIGNEES> and </ASSIGNEES> labels, and the <AGENTS> and </AGENTS> labels should be interpreted as the keywords labeling this document. Moreover, the SGML labels identifying each keyword are also interpreted as higher-level entities to be used in the document label taxonomy (discussed further in Section 4). Thus, for example, each string found between <INVENTOR> and </INVENTOR> will appear beneath a node labeled "INVENTOR" in the

initial taxonomy. Finally, in case it is useful to the knowledge discovery operation, the tags file also informs Document Explorer that the date of the document can be found between the <ISSUED> and </ISSUED> labels. All other SGML labels (such as <A>) are ignored.

```
DOC DOCUMENT
TITLE SUBJECT
BODY CLAIMS ABSTRACT
DATE ISSUED
TAGS INVENTORS ASSIGNEES
AGENTS
```

Figure 4. Tags file for the patent-records collection.

To show the flexibility this format provides, we also include an example of a document from the Reuters newswire domain in Figure 5, and the corresponding tags file in Figure 6. In this case each document in a file are labeled with a <REUTERS> </REUTERS> pair (note that the additional annotations in the <REUTERS> label are ignored for this processing), "earn" and "acq" are defined as keywords that will appear under the TOPICS node in the keyword taxonomy, and so on. Empty keyword tags, such as for PEOPLE, ORGS, EXCHANGES, and COMPANIES are ignored, as are SGML tags (such as "<UNKNOWN>" that are not defined in the tags file).

```
<REUTERS>
<DATE>26-FEB-1987
15:19:15.45</DATE>
<TOPICS><D>earn</D><D>acq</
D></TOPICS>
<PLACES><D>usa</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<TEXT>
<TITLE>OHIO        MATTRESS
&lt;OMT> MAY HAVE LOWER 1ST
QTR NET</TITLE>
<BODY>Ohio Mattress Co said
its first quarter, ending
February 28, profits may be
below the 2.4 mln dlrs, or
15 cts a share, earned in
the first quarter of fiscal
1986.
    The company said any
decline would be due to
expenses related to the
acquisitions in the middle
of the current quarter of
seven licensees of Sealy
Inc.
Reuter
</BODY>
</TEXT>
</REUTERS>
```

Figure 5. An example Reuters newswire document.

```
DOC REUTERS
TITLE TITLE
BODY BODY
TAGS TOPICS PLACES PEOPLE
ORGS EXCHANGES COMPANIES
DATE DATE
```

Figure 6. The tags file for Reuters documents.

3 Linguistic Preprocessing

Once received by Document Explorer, a number of well-established linguistic preprocessing steps are performed on each resulting document. First, it performs tokenization, by which white space and punctuation are used to identify the lexical items in the text. Next, Document Explorer performs part-of-speech tagging, which automatically associates morpho-syntactic categories such as *noun*, *verb*, *adjective*, etc., to the words in the document. Document Explorer performs such tagging using a rule-based approach similar to the one presented by Brill (1995), which is known to yield satisfying results (96% accuracy) provided that a large lexicon and some manually hand-tagged data is available for training. Finally, Document Explorer performs lemmatization (Hull, 1996), a linguistically better-founded version of stemming, in which the root portion of each word is identified.

3.1 Term Extraction

The term extraction module is responsible for labeling each document with a set of terms extracted from the document. Figure 1 gave an example of the results of this process on an excerpt of a document taken from an article published by the Reuters news service on 12 May 1996. Terms in this excerpt that were identified and designated as interesting by the term extraction module are underlined. Thus, for example, "profit" and "Canadian Imperial Bank of Commerce" are both extracted terms that would be used to label this document. This section describes the two components of Document Explorer's text extraction module: term generation, and term filtering.

3.2 Term Generation

In the term generation stage, sequences of tagged lemmas are selected as potential term candidates on the basis of relevant morpho-syntactic patterns (such as "Noun Noun", "Noun Preposition Noun", "Adjective Noun", etc.). The candidate combination stage is performed in several

passes. In each pass, an *association coefficient* (to be defined shortly) between each pair of adjacent terms is calculated and any pair whose association coefficient is large enough is combined. In the case of competing possibilities involving overlapping terms (such as (t_1 t_2) and (t_2 t_3) in (t_1 t_2 t_3)), the pair having the better association coefficient is replaced first. The documents are then updated by converting all combined terms into a new single term and the whole procedure is then repeated until no new terms are generated.

The nature of the patterns used for candidate generation is an open research question. Daille (1994, 1996) proposed specific operators (such as overcomposition, modification, and coordination) to select longer terms as combinations of shorter ones. Justeson and Katz (1995) suggest accepting prepositions as well as adjectives and nouns. This approach generate a much larger number of term results; Frantzi (1997) only accepts (Noun|Adjective)-Noun sequences to reduce the amount of "bad" terms.

In Document Explorer we used two basic patterns: Noun-Noun and Adjective-Noun, but we also allowed the insertion of any kind of Determiner, Preposition or Subordinating Conjunction. Therefore sequences such as "health program for the elderly", "networking software for personal computers", "operating system of a computer" or "King Fahd of Saudi Arabia" are accepted as well.

To compute the association coefficient for combining two terms we currently use an *ad hoc* co-occurrence metric that computes a function of the number of times that the two terms match the possible extraction patterns. The term generation process combines two terms into a bigger term only if the value of this coefficient is over a threshold, T_{freq}. Although Document Explorer provides a default value for this threshold, it was designed so that a user can vary the threshold to affect the term generation process. For the experiments reported later a fixed value of 8 was used.

3.3 Term Filtering

The term generation stage produces a set of terms associated with each document without taking into account the relevance of these terms in the framework of the whole document collection. We therefore allow the term generation stage to create more terms than is truly desired, complementing generation with an additional filtering stage that prunes generated terms based on their frequencies of occurrence throughout the collection. For example, the following are examples of two-word terms that were identified during term generation, but were later eliminated during term filtering in one sample text-mining session: `right direction`, `other issue`, `point of view`, `long way`, `question mark` and `same time`.

Our goal in term filtering is to identify terms that may not to be of interest in the context of the whole document collection either because they do not occur frequently enough or because they occur in a fairly constant distribution among the different documents. Our approach uses a statistical relevance-scoring function that assigns a score to each generated term based on their occurrence patterns in the collection, and the top M (for a user-specified M) are taken as the final set of terms to be used in text mining.

As in term generation, Document Explorer allows a user to select and combine these filtering methods if the user desires such control over the term generation process. For the experiments given later only the tf-idf –based filter was used, with a fixed threshold of 4.5.

4 Taxonomy Construction

One of the crucial issues in performing text mining at the term level is the need for a term taxonomy. Even with filtering, there are often a very large number of generated terms (10,000 is not unusual), and knowledge discovery processes often rely on some hierarchy of terms so as to form results at a higher level of granularity. Thus, for example, having a term taxonomy would enables the production of general association rules (Srikant and Agrawal, 1995). These rules capture relationships between *groups* of terms rather than individual terms. A taxonomy is also important in other text mining algorithms such as Maximal Association Rules and Frequent Maximal Sets (Feldman et al, 1997).

A taxonomy also enables the user to specify mining tasks in a concise way. For instance when trying to generate association rules, rather then looking for all possible rules, the user can specify interest only in the relationships of companies in the context of business alliances. In order to do so, we need two nodes in the term taxonomy marked "business alliances" and "companies". The first node would contain terms related to business alliances such as "joint venture", "strategic alliance", "combined initiative", etc., while the second node is the parent of all company names in our system (which could be the result of human effort specifying such a higher-level term, but in our application we used a set of rules and knowledge extracted from WWW directories to generate company names).

Building a term taxonomy is a time consuming task. We therefore provide a user with a set of tools for semi-automatic construction of such a taxonomy. As was already discussed in Section 2, the process of specifying the meaning of the SGML mark-ups in a tags file gives an

initial hierarchy for any pre-existing keyword labels. To generate a taxonomy for extracted terms our main tool is the taxonomy editor, depicted in Figure 8. This tool enables the user to read a set of terms or an external taxonomy, and use them to update the system's term taxonomy. It also allows simple operations, such as saying that "Apple Computer Corp" and "Apple Computer Inc" should be under a single node. The user can also drag and move entire subtrees in creating and modifying the taxonomy. Finally, the user can also specify a set of terms via regular to identify sets of terms to be placed together when defining or modifying higher-level terms in the taxonomy. For example, Figure 8 shows the terms found when specifying the pattern *petroleum**.** The initial taxonomy depicted on the left in this figure contains all terms extracted from the Reuters 52,000 document collection (shown in the left tree). The terms matching the query are shown in the middle tree and the right tree is the taxonomy being created.

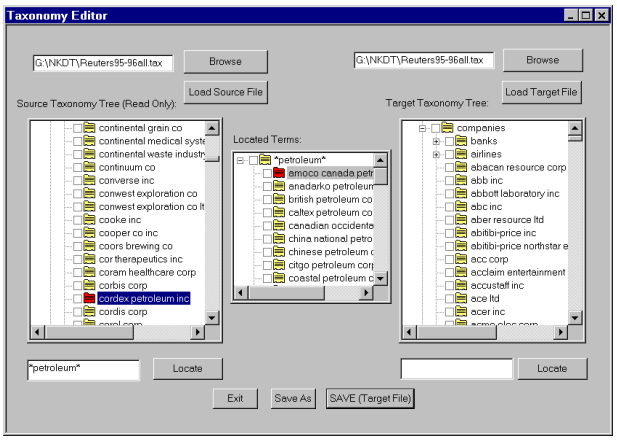

Figure 8. Taxonomy Editor

The taxonomy editor also includes a semi-automatic tool for taxonomy editing called the taxonomy refiner. The taxonomy refiner compares generated frequent sets – terms that often co-occur – against the term taxonomy. When most of the terms of a frequent set are determined to be siblings in the taxonomy hierarchy the tool suggests adding the remaining terms as siblings as well.

For example, if our taxonomy currently contains 15 companies under the "tobacco companies" and the system generated a frequent set containing many tobacco companies, one of which does not appear in the taxonomy, the taxonomy refiner will suggest adding this additional company to the taxonomy as a tobacco company (e.g., Phillip Morris). The term refiner also has a term clustering module again suggest that terms clustered together be placed as siblings in the taxonomy.

5 Results

In this section we show examples of the use of Document Explorer, and some evaluation of its performance. For most of what follows use 51,725 documents from the Reuters financial news for the years 1995-1996. This collection is 120M in size and contains over 170,000 unique words. Each document contained on average 864 words. In the term generation stage, 1.25M terms were identified, 154K of them unique. After term filtering we were left with 975K terms (approximately 45 terms per document), 16,847 of them unique. Thus, even though we expand beyond words to multi-word terms, the resulting set of terms was reduced in size by more than a factor of 10.

Figure 9 gives an example of a user requesting associations between companies and business alliances mentioned in the various documents. The user thus constrains the left-hand side (LHS) of the association to contain extracted terms that occur under the **companies** node in the hierarchy, and the right-hand side (RHS) to contain terms that occur under the **business alliances** node.

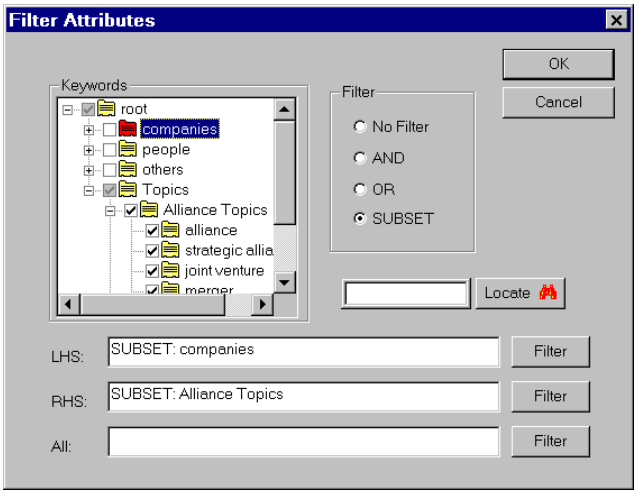

Figure 9. Specifying a filter to generate association rules with children of the Companies node on the left-hand side of the rule and children of Alliance Topics on the right.

Using the Reuters document corpus described above, Document Explorer generated 12,000 frequent sets complying with the restriction specified by this association-rule query (using a support threshold of 5 documents and confidence threshold of 0.1). These frequent sets generated 575 association. A sample of these rules is presented in Figure 10, where the numbers presented at the end of each rule are the rule's support and confidence.

```
america online inc, bertelsmann ag ⇒ joint venture 13/0.72
apple computer inc, sun microsystems inc ⇒ merger talk 22/0.27
apple computer inc, taligent inc ⇒ joint venture 6/0.75
sprint corp, tele-communications inc ⇒ alliance 8/0.25
burlington northern inc, santa fe pacific corp ⇒ merger 9/0.23
lockheed corp, martin marietta corp ⇒ merger 14/0.4
chevron corp, mobil corp ⇒ joint venture 11/0.26
intuit inc, novell inc ⇒ merger 8/0.47
bank of boston corp, corestates financial corp ⇒ merger talk 7/0.69
```

Figure 10. A sample of the association rules found by Document Explorer that comply with the restrictions specified in Figure 9.

The example above illustrates the advantages of performing text mining at the term level. Terms such as **joint venture** would not appear if working at the word level, nor would company names, such as **santa fe pacific corp** and **bank of boston corp**. It also demonstrates the utility of the term taxonomy in specifying the association-rule query.

Figure 11 shows a different use for terms in knowledge discovery, through its use in tools that permit interactive browsing of a collection. The figure depicts the use of a tool that allows browsing based on term distributions. The given knowledge-discovery session started by computing the distribution of all alliance-related topics, i.e., topics under the **alliance** node in the taxonomy. Upon finding that the most frequent topic was **joint venture**, the user then computed the **company** distribution of that topic, **international business machines** (an extracted term) was the company that co-occurred the most with **joint venture**.

The user then chose to compute the company distribution of **mci communication corp** (in the context of **joint venture**), finding that **sprint** was the company with the highest frequency. Finally, the user then chose to compute the people distribution of News Corp in the context of “join venture” and “MCI Communication Corp”.

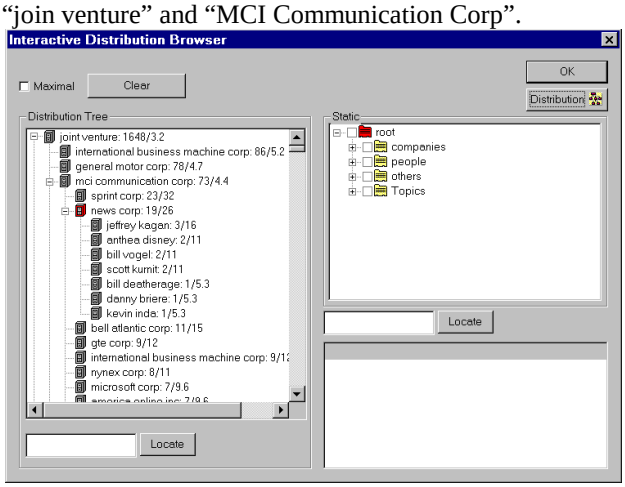

Figure 11. Interactive exploration of term distributions.

Document Explorer provides also a set of visual maps that depict the relationship between entities in the corpus. The context graph shown in Figure 12 depicts the relationship between "companies" in the context of “joint venture”. In many cases, the number of edges in the graph is too large. Hence we provide a filter mechanism that enables the user to see only edges that exceed a given threshold (together with the adjacent nodes). In Figure 12 the user picked a threshold of 15. The weights of the edges (number of documents in which the nodes appear in the context of “joint venture”) are noted alongside the edge.

The graph clearly exposes the main industry clusters, which are shown as disconnected components of the graph: a telephony industry cluster, the internet provider/broadcasting cluster, automobile companies cluster, entertainment cluster and a chemical companies cluster. The Context Graph provides a powerful way to visualize relationship encapsulated in thousands of documents.

The map in Figure 13, shows the connections between "People", "Brokerage Houses" and "Computer Companies", with respect to "mergers". Color-coded connection-lines represent the strength of the connection.

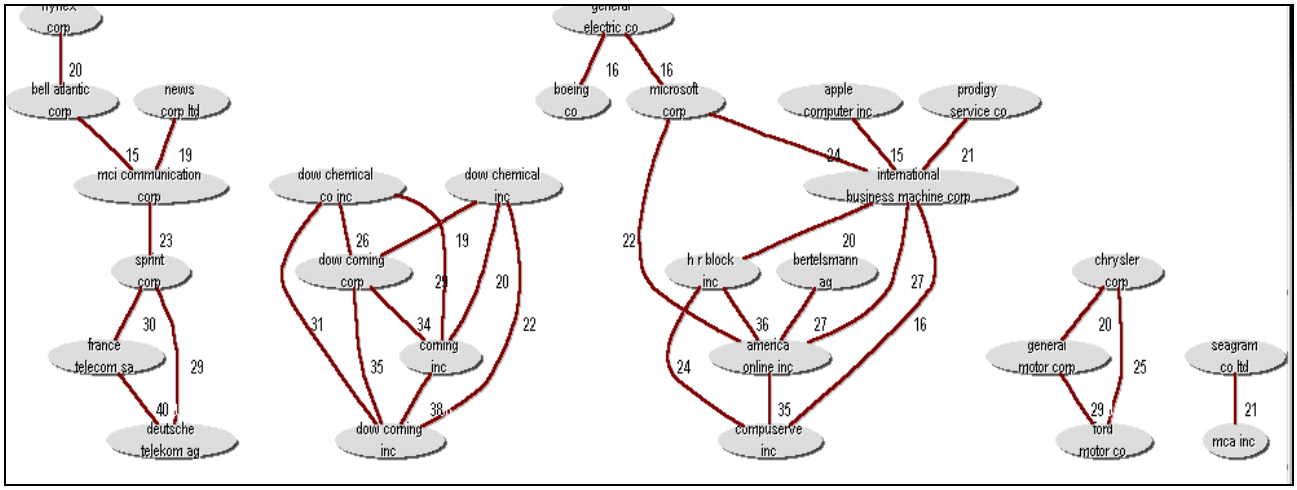

Figure 12 – Context Graph (Companies in Context of "Joint Venture")

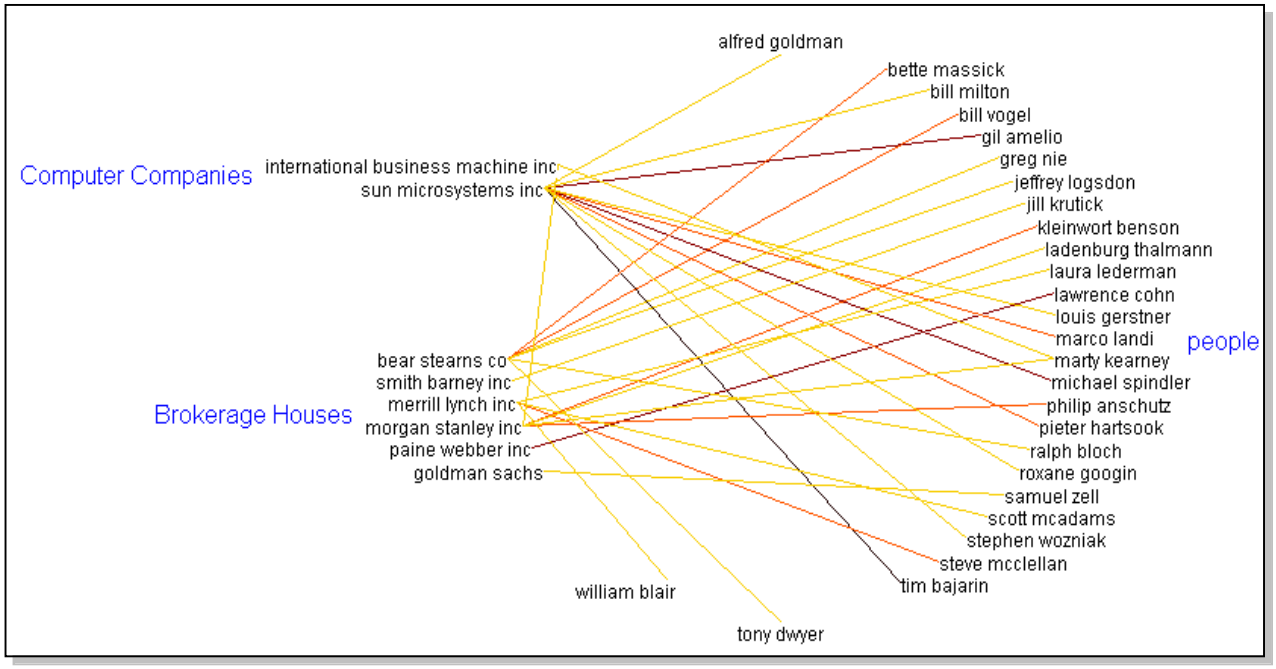

Figure 13 – Category Graph (relationship between People, Brokerage Houses and Computer Companies in Context of "merger")

6 Summary

Previous approaches to text mining have assumed that documents were labeled either with higher-level keywords, or simply the set of words occurring in the documents. In the case of keywords, labels were either assigned manually, as is done by some on-line information services (e.g., Dialog and Reuters), which is a very expensive and time-consuming process, by having the user specify keyword labeling procedures, which are often inaccurate and difficult to specify, or via machine-learning algorithms, which are also inaccurate and require some initial amount of human labeling for each new keyword. Further, the limited number of keywords that are typically used constrained the amount of information that is represented about each document, and thus that can be analyzed by knowledge-discovery operations. On the

other hand, systems that perform knowledge discovery on the words contained in each document tend to produce a huge number of often meaningless results. Further, the large number of word that must be considered often leads to unreasonable execution times and the memory requirements.

Text mining at the term level attempts to hit a midpoint, reaping some benefits from each of these extremes while avoiding many of their pitfalls. On the one hand, there is no need for human effort in labeling documents, and we are not constrained to a smaller set of labels that lose much of the information present in the documents. Thus the system has the ability to work on new collections without any preparation, as well as the ability to merge several distinct collections into one (even though they might have been tagged according to different guidelines which would prohibit their merger in a tagged based system). On the other hand, the number of meaningless results is greatly reduced and the execution time of the mining algorithms is also reduced relative to pure word-based approaches.

Text mining at the term level thus hits a useful middle ground on the quest for tools for understanding the information present in the large amount of data that is only available in textual form. Text Mining at the term level serves as a powerful technique to manage knowledge encapsulated in large document collections.

7 References

Anand T. and Kahn G., 1993. Opportunity Explorer: Navigating Large Databases Using Knowledge Discovery Templates. In *Proceedings of the 1993 workshop on Knowledge Discovery in Databases*.

Bookstein A., Klein S.T. and Raita T., 1995. Clumping Properties of Content-Bearing Words. In *Proceedings of SIGIR'95*.

Brachman R. J., Selfridge P. G., Terveen L. G., Altman B., Borgida A., Halper F., Kirk T., Lazar A., McGuinness D. L., and Resnick L. A., 1993. Integrated Support for Data Archaeology. *International Journal of Intelligent and Cooperative Information Systems* 2(2):159-185.

Brill E., 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging, *Computational Linguistics*, 21(4):543-565

Church K.W. and Hanks P., 1990. Word Association Norms, Mutual Information, and Lexicography, *Computational Linguistics*, 16(1):22-29

Cohen W. and Singer Y., 1996. Context Sensitive Learning Methods for Text categorization. In *Proceedings of SIGIR'96*.

Daille B., Gaussier E. and Lange J.M., 1994. Towards Automatic Extraction of Monolingual and Bilingual Terminology, In *Proceedings of the International Conference on Computational Linguistics*, COLING'94, pages 515-521.

Dunning T., 1993. Accurute Methods for the Statistics of Surprise and Coincidence, *Computational Linguistics*, 19(1).

Feldman R., and Hirsh H., 1996. Exploiting Background Information in Knowledge Discovery from Text. *Journal of Intelligent Information Systems*. 1996.

Feldman R., Aumann Y., Amir A., Klösgen W. and Zilberstien A., 1997. Maximal Association Rules: a New Tool for Mining for Keyword co-occurrences in Document Collections, In *Proceedings of the 3rd International Conference on Knowledge Discovery*, KDD-97, Newport Beach, CA.

Feldman R. and Dagan I., 1995. KDT – Knowledge Discovery in Texts. In *Proceedings of the First International Conference on Knowledge Discovery*, KDD-95.

Frantzi T.K., 1997. Incorporating Context Information for the Extraction of Terms. In *Preceedings of ACL-EACL'97*.

Frawley W. J., Piatetsky-Shapiro G., and Matheus C. J., 1991. Knowledge Discovery in Databases: an Overview. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 1-27, MIT Press.

Hahn, U.; and Schnattinger, K. 1997. Deep Knowledge Discovery from Natural Language Texts. In *Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining*, 175-178.

Hull D., 1996. Stemming algorithms - a case study for detailed evaluation. *Journal of the American Society for Information Science*. 47(1):70-84.

Justeson J. S. and Katz S. M., 1995. Technical Terminology: Some linguistic properties and an algorithm for identification in text. In *Natural Language Engineering*, 1(1):9-27.

Klösgen W., 1992. Problems for Knowledge Discovery in Databases and Their Treatment in the Statistics Interpreter EXPLORA. *International Journal for Intelligent Systems*, 7(7):649-673.

Lent B., Agrawal R. and Srikant R., 1997. Discovering Trends in Text Databases. In *Proceedings of the 3rd International Conference on Knowledge Discovery*, KDD-97, Newport Beach, CA.

Piatetsky-Shapiro G. and Frawley W. (Eds.), 1991. Knowledge Discovery in Databases, AAAI Press, Menlo Park, CA.

Rajman M. and Besançon R., 1997. Text Mining: Natural Language Techniques and Text Mining Applications. In *Proceedings of the seventh IFIP 2.6 Working Conference on Database Semantics* (DS-7), Chapam & Hall IFIP Proceedings serie. Leysin, Switzerland, Oct 7-10, 1997.

Salton G. and Buckley C., 1988. Term-weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*. 24(5):513-523.

Srikant R. and Agrawal R., 1995. Mining generalized association rules. In *Proceedings of the 21st VLDB Conference*, Zurich, Swizerland.