Text Mining via Information Extraction

Ronen Feldman , Yonatan Aumann, Moshe Fresko,
Orly Liphstat, Binyamin Rosenfeld, Yonatan Schler

Department of Mathematics and Computer Science
Bar-Ilan University
Ramat-Gan, ISRAEL
Tel: 972-3-5326611
Fax: 972-3-5326612
feldman@cs.biu.ac.il

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 via information extraction,* in which knowledge discovery takes place on a more focused collection of events and phrases that are extracted from and label each document. These events plus additional higher-level entities are then organized in a hierarchical taxonomy and are used in the knowledge discovery process. This approach was implemented in the Textoscope system. Textoscope consists of a document retrieval module which converts retrieved documents from their native formats into SGML documents used by Textoscope; an information extraction engine, which is based on a powerful attribute grammar which is augmented by a rich background knowledge; a taxonomy-creation tool by which the user can help specify higher-level entities that inform the knowledge-discovery process; and a set of knowledge-discovery tools for the resulting event-labeled documents. We evaluate our approach on a collection of newswire stories extracted by Textoscope's own agent. Our results confirm that Text Mining via information extraction serves as an accurate and powerful technique by which to manage knowledge encapsulated in large document collections.

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. 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. 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 unstructured text. This paper focuses on the problem of *text mining*, performing knowledge discovery from collections of unstructured text. One common technique [3,4,5] has been to assume that associated with each document is a set of labels and then 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 indicates 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 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 [6] 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).

In this paper we instead present a middle ground, in which we perform *Information extraction* on each document to find events and entities that are likely to have meaning in the domain, and then perform mining on the extracted events labeling each document. Unlike word-based approaches, the extracted events are fewer in number and tend to represent more meaningful concepts and relationships in the domain of the document. A possible event can be that a company did a joint venture with a group of companies or that a person took position at a company. Unlike keyword approaches, our information-extraction method eliminates much of the difficulties in labeling documents when faced with a new collection or new keywords. While we rely on a generic capability of recognizing proper names which is mostly domain-independent, when the system is to be used in new domains, some work is needed for defining additional event schemas. Textoscope provides a complete editing/compiling/debugging environment for defining the new event schemas. This environment enables easy creation and manipulation of information extraction rules.

This paper describes Textoscope, a system that embodies this approach to *text mining via information extraction.* The overall structure of Textoscope is shown in Figure 1. The first step is to convert documents (either internal documents or external documents fetched by using the Agent) into an SGML format understood by Textoscope. 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 Textoscope's SGML format. Finally, KDD operations are performed on the event-labeled documents.

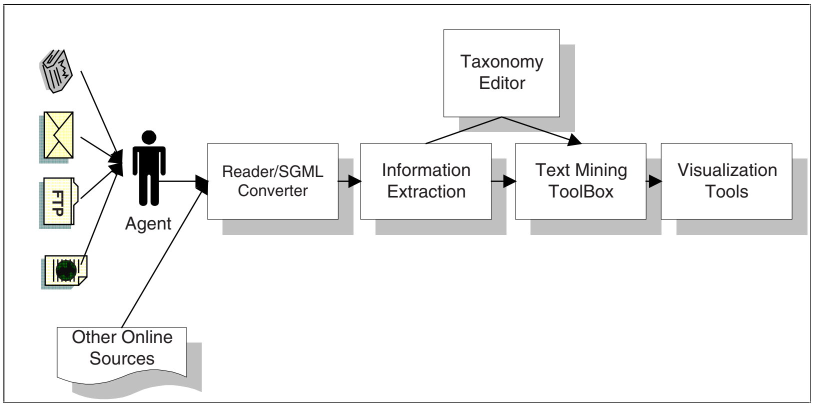


Fig. 1. Textoscope architecture.

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 Textoscope's various components. The the linguistic preprocessing steps, Textoscope's Information extraction engine, its tool for creating a taxonomic hierarchy for the extracted events, and, finally, a sample of its suite of text mining tools. We give examples of mining results on a collection of newswire stories fetched by our agent.

2 Information Extraction

Information Extraction (IE) aims at extracting instances of predefined templates from textual documents. IE has grown to be a very active field of research thanks to the MUC (Message Understanding Conference) initiative. MUC was initiated by DARPA in the late 80's in response to the information overload of on-line texts. One of the popular uses of IE is proper name extraction, i.e., extraction of company names, personal names, locations, dates, etc. The main components of an IE system are tokenization, zoning (recognizing paragraph and sentence limits), morphological and lexical processing, parsing and domain semantics [1,7]. Typically, IE systems do not use full parsing of the document since that is too time consuming and error prone. The methods typically used by IE systems are based on shallow parsing and use a set of predefined parsing rules. This "knowledge-based" approach may be very time consuming and hence a good support environment for writing the rules is needed.

Textoscope preprocesses the documents by using its own internal IE engine. The IE engine makes use of a set of predefined extraction rules. The rules can make use of a rich set of functions that are used for string manipulation, set operations and taxonomy construction. We have three major parts to the rules file. First we define all the events that we want to extract from the text. An example of an event is "Company1 Acquired Company2", or "Person has Position in Company". The second part are word classes, collections of words that have a similar semantic property. Examples of word classes are company extensions (like "inc", "corporation" "gmbh" "ag" etc.) and a list of common personal first names. The third and last part are rules that are used to extract events out of the documents. There are two types of rules, event-generation rules and auxiliary rules.

Each event-generating rule has three parts, a pattern, a set of constraints (on components of the pattern), and a set of events that are generated from the pattern. An auxiliary rule contains just a pattern. The system supports three types of patterns, AND-patterns , sequential pattern (which has a similar semantics to a prolog DCG rule), and skip patterns. Skip patterns enable the IE engine to skip a series of tokens until a member of a word class is found.

Here is an example of an event generating rule that uses an auxiliary rule:

```
@ListofProducts = ( @ProductList are [ registered ] trademarks of @Company
@! )
                        > ProductList: Products = 0.
@ProductList = ( @Product , @ProductList1 @!).
@ProductList1 = ( @Product, @ProductList1 @!).
@ProductList1 = ( @Product [ , ] and @Product @! ).
```

In this case we look for a list of entities that is followed by the string "are registered trademarks" or "are trademarks". Each of the entities must conform to the syntax of a @Product.

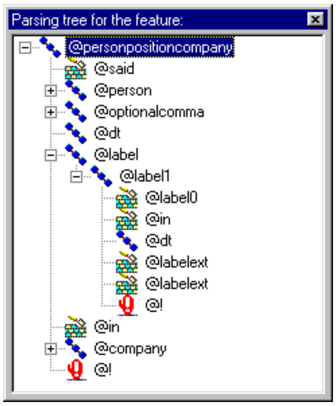
We have used many resources found on the WWW to acquire lists of common objects such as countries, states, cities, business titles (e.g., CEO, VP of Product Development, etc.), technology terms etc. Technology terms for instance were extracted from publicly available glossaries. We have used our IE engine (with a specially designed rule-set) to automatically extract the terms from the HTML source of the glossaries. In addition, we have used words lists of the various part of speech categories (nouns, verbs, adjectives, etc.). These word lists are used inside the rules to direct the parsing.

Each document is processed using the IE engine and the generated events are inserted into the document repository. In addition to the events inserted, each document is annotated with terms that are generated by using term extraction algorithms [2,5]. This enables the system to use co-occurrence between terms to infer relations that were missed by the IE engine. The user can select the granularity level of the co-occurrence computation, either document level, paragraph level or sentence level. Clearly, if the granularity level is selected to be document-level, the precision will decrease, while the recall will increase. On the other hand, selecting a sentence-level granularity will yield higher precision and lower recall. The default granularity level is the sentence level, terms will be considered to have relationship only if they co-occur within the same sentence. In all the analysis modules of the Textoscope system the user can select whether relationships will be based solely on the events extracted by the IE engine, on the term extraction, or a combination of two.

One of the major issues that we have taken into account while designing the IE Rule Language was allowing the specification of common text processing actions within the language rather than resorting to external code written in C/C++. In addition to recognizing events, the IE engine allows the additional analysis of text fractions that were identified as being of interest. For instance, if we have identified that a given set of tokens is clearly a company name (by having as a suffix one of the predefined company extensions), we can insert into a dynamic set called DCompanies the full company name and any of its prefixes that still constitute a company name. Consider the string “Microsoft Corporation”, we will insert to DCompanies both “Microsoft Corporation”, and “Microsoft”. Dynamic sets are handled at five levels: there are system levels sets, there are corpus level sets, there are document level sets, paragraph level sets and sentence level sets. System level sets enable knowledge transfer between corpuses, while corpus level sets enable knowledge transfer between documents in the same corpus. Document level sets are used in cases where the knowledge acquired should be used just for the analysis of the rest of the document and it is not applicable to other documents. Paragraph and sentence level sets are used in discourse analysis, and event linking.

The IE engine can learn the type of an entity by the context in which the entity appears. As an example, consider a list of entities some of which are unidentified. If the engine can determine the type of at least one of them, then the types of all other entities are determined to be the same. For instance, given the string “joint venture among affiliates of Time Warner, MediaOne Group, Microsoft, Compaq and Advance/Newhouse.”, since the system has already identified Microsoft as being a company, it determined that Time Warner, MediaOne Group, Compaq and Advance/Newhouse are companies as well. The use of the list-processing rules provided a considerable boost the accuracy of the IE engine. For instance, in the experiment described in Section 4, it caused recall to increase from 82.3% to 92.6% while decreasing precision from 96.5% to 96.3%.

Textoscope provides a rich support environment for editing and debugging the extraction rules. On the editing front, Textoscope provides a visual editor for building the rules that enables the user to create rules without having to memorize the exact syntax. On the debugging front, Textoscope provides two main utilities. First, it provides a visual tool that enables one to see all the events that were extracted from the document. The user can click on any of the events and then see the exact text where this event was extracted from. In addition the system provides an interactive derivation tree of the event, so that the user can explore exactly how the event was generated. An example of such a derivation tree is shown in Figure 2. Here we parsed the sentence "We see the Nucleus Prototype Mart as the missing link to quickly deploying high value business data warehouse solutions, said David Rowe, Director of Data Warehousing Practice at GE Capital Consulting", and extracted the event that David Rowe is the Director of Data Warehousing Practice at a company called GE Capital Consulting. Each node in the derivation tree is annotated by an icon that symbolizes the nature of the associated grammar feature. The second debugging tool provides the user with the ability to use a tagged training set and rate each of the rules according to their contribution to the precision and recall of the system. Rules that cause precision to be lower and do not contribute towards a higher recall can be either deleted or modified.



```
Parsing tree for the feature:
⊟ @personpositioncompany
    @said
  ⊞ @person
  ⊞ @optionalcomma
    @dt
  ⊟ @label
    ⊟ @label1
        @label0
        @in
        @dt
        @labelext
        @labelext
        @!
    @in
  ⊞ @company
    @!
```

Fig. 2. An Interactive Derivation Tree of an Extracted Event

The second debugging tool provides the user with the ability to use a tagged training set and rate each of the rules according to their contribution to the precision and recall of the system. Rules that cause precision to be lower and do not contribute towards a higher recall can be either deleted or modified.

The events that were generated by the IE engine are used also for the automatic construction of the taxonomy. Each field in each of the events is used as a source of values for the corresponding node in the taxonomy. For instance, we use the Company field from the event "Person, Position, Company" to construct the Company node in the taxonomy. The system contains several meta rules that enable the construction of a multi-level taxonomy. Such a rule can be, for instance, that Banks are Companies and hence the Bank node will be placed under the Company node in the Taxonomy.

Textoscope constructs a thesaurus that contains lists of synonyms. The thesaurus is constructed by using co-reference and a set of rules for deciding that two terms actually refer to the same entity. Example of a synonym list that is constructed by the system is { "IBM", "International Business Machines Corp" and "Big Blue" }. Textoscope also includes a synonym editor that enables the user to add/modify/delete synonym lists. This enables the user to change the automatically created thesaurus and customize it to her own needs.

3 Results

We tested the accuracy of the IE engine by analyzing collections of documents that were extracted by the Agent from MarketWatch.com. We started by extracting 810 articles from MarketWatch.com which mentioned "ERP". We have created 30 different events focused around companies, technologies, products and alliances. We have defined more than 250 word classes and have used 750 rules to extract those 30 event types. The rule scoring tool described in Section 3 was proved to be very useful in the debugging and refinement of the rule set. After the construction of the initial rule set we were able to achieve an F-Score of 89.3%. Using the rule scoring utility enabled us to boost the F-Score to 96.7% in several hours.

In order to test the rule set, we have used our agent again to extract 2780 articles that mentioned "joint venture" from MarketWatch.com. We were able to extract 15,713 instances of these events. We have achieved a 96.3 precision and 92.6 recall on the company, people, technology and product categories and hence an F-Score of 94.4% ($\beta = 1$) where $F = \frac{(\beta^2+1)PR}{(\beta^2 P+R)}$. These results are in par with the results achieved by the FASTUS system [1] and the NETOWL system (www.netowl.com).

We will now show how Textoscope enables us to analyze the events and terms that were extracted from the 2780 articles. Textoscope provides a set of visual maps that depict the relationship between entities in the corpus. The context graph shown in Figure 3 depicts the relationship between "technologies". The weights of the edges (number of documents in which the technologies appear in the same context) are coded by the color of the edge, the darker the color, the more frequent the connection. The graph clearly reveals the main technology clusters, which are shown as disconnected components of the graph: a security cluster and internet technologies cluster. We can see strong connections between electronic commerce and internet security, between ERP and data warehousing, and between ActiveX and internet security.

In Figure 4, we can view some of the company clusters that were involved in some sort of alliance ("joint venture", "strategic alliance", "commercial alliance", etc.).

The Context Graph provides a powerful way to visualize relationship encapsulated in thousands of documents.

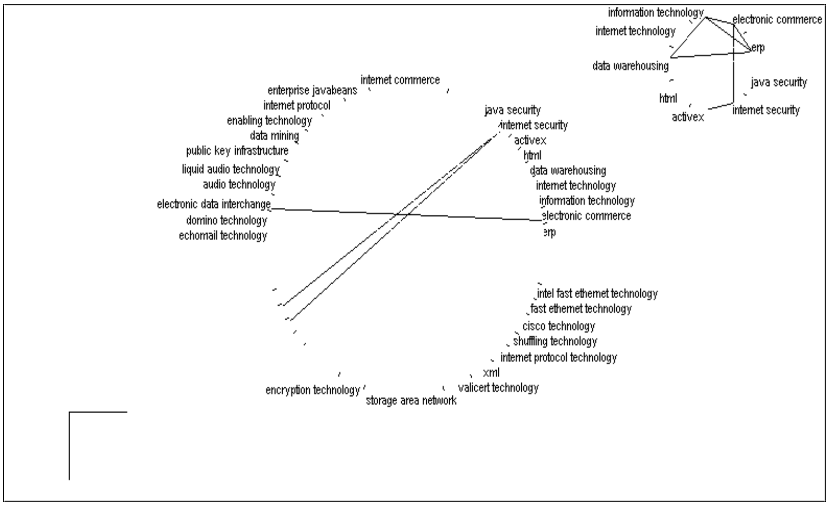


Fig. 3. Context Graph (technologies)

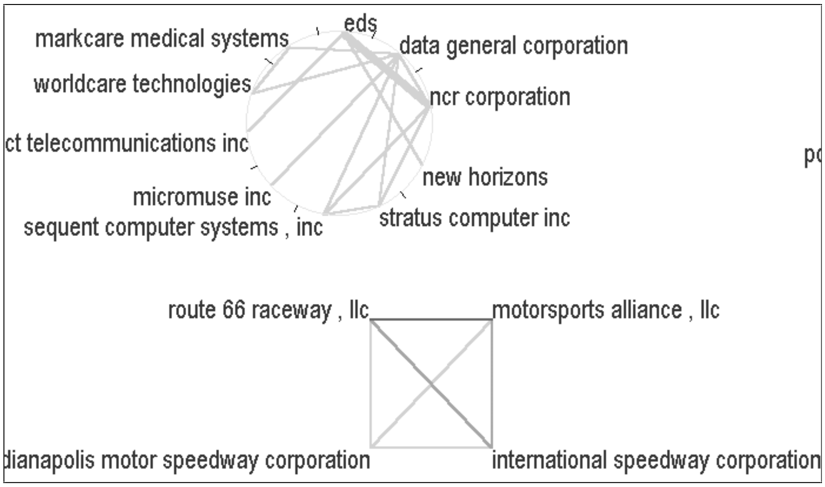


Fig. 4. Joint Venture Clusters

4 Summary

Text mining based on Information Extraction 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 using Information Extraction 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. The powerful combination of precise analysis of the documents and a set of visualization tools enable the user to easily navigate and utilize very large document collections.

References

1. Appelt, Douglas E., Jerry R. Hobbs, John Bear, David Israel, and Mabry Tyson, 1993. ‘‘FASTUS: A Finite-State Processor for Information Extraction from Real-World Text”, *Proceedings*. IJCAI-93, Chambery, France, August 1993.

2. 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.

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

4. 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.

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

6. 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.

7. Soderland S., Fisher D., Aseltine J., and Lehnert W., "Issues in Inductive Learning of Domain-Specific Text Extraction Rules," Proceedings of the Workshop on New Approaches to Learning for Natural Language Processing at the Fourteenth International Joint Conference on Artificial Intelligence, 1995.