Discovering Trends in Text Databases

Brian Lent* and **Rakesh Agrawal** and **Ramakrishnan Srikant**
IBM Almaden Research Center
San Jose, California 95120, U.S.A.
lent@cs.stanford.edu, {ragrawal,srikant}@almaden.ibm.com

Abstract

We describe a system we developed for identifying trends in text documents collected over a period of time. Trends can be used, for example, to discover that a company is shifting interests from one domain to another. Our system uses several data mining techniques in novel ways and demonstrates a method in which to visualize the trends. We also give experiences from applying this system to the IBM Patent Server, a database of U.S. patents.

Introduction

We address the problem of discovering trends in text databases. We are given a database $\mathcal{D}$ of documents. Each document consists of one or more text fields and a timestamp. The unit of text is a *word* and a *phrase* is a list of words. (We defer the discussion of more complex structures till the "Methodology" section.) Associated with each phrase is a *history* of the frequency of occurrence of the phrase, obtained by partitioning the documents based upon their timestamps. The frequency of occurrence in a particular time period is the number of documents that contain the phrase. (Other measures of frequency are possible, e.g. counting each occurrence of the phrase in a document.) A *trend* is a specific subsequence of the history of a phrase that satisfies the users' query over the histories. For example, the user may specify a "spike" query to finds those phrases whose frequency of occurrence increased and then decreased.

Approach

Our approach to the problem of discovering trends is to reuse as much of the current data mining technology as possible. In doing so, we have two major mining components to the system: phrase identification using sequential patterns mining (Agrawal & Srikant 1995; Srikant & Agrawal 1996) and trend identification using shape queries (Agrawal *et al.* 1995). We begin by cleansing and parsing the input data, and separating the documents based on their timestamps. We then assign a transaction ID to each word of every document treating the words as items in the data mining algorithms (the details of this assignment are discussed in the "Methodology" section). This transformed data is then mined for dominant words and phrases, and the results saved. The user's query is translated into a shape query and this query is then executed over the mined data yielding the desired trends. The final step in the process is to visualize the results.

Related Work

An approach to discovering interesting patterns and trend analysis on text documents was presented in (Feldman & Dagan 1995). The text is first annotated with a set of concepts, organized as a hierarchy. Treating the concept hierarchy as a distribution of probabilities, they identify several model distributions to which a given concept hierarchy can be compared. Interesting concepts are those that differ from their model distribution. Analyzing trends involves the comparison of concept distributions using old data with distributions using new data.

In (Feldman & Hirsh 1996), the authors find associations between the keywords or concepts labeling the documents using background knowledge about relationships among the keywords. The purpose of the knowledge base is to supply unary or binary relations amongst the keywords labeling the documents.

Using words and phrases to describe themes and concepts in text documents has been studied by the information retrieval community. The work on Latent Semantic Indexing (LSI) (Deerwester *et al.* 1990) describes a mathematical model of relating word associations as weighted vectors that represent "concepts" found within the documents. Using LSI, a query can retrieve a document even when they share no words, but do share a similar concept. However, building the model takes $O(tk^4d)$ time, where t is the number of

* Current address: Department of Computer Science, Stanford University. Continuing support has been provided by a graduate fellowship of the Department of Defense Office of Naval Research.

terms or words, k is the number the major concepts in the model (typically defined from 100 to 300), and d is the number of documents.

The use of phrases to build more advanced queries is discussed in (Croft, Turtle, & Lewis 1991). In this work, the authors identify phrases as concepts and as relationships between concepts. The usefulness of phrases is shown in (Lewis & Croft 1990) where the quality of text categorization is improved by using word clusters and phrases. The reliability of using words and phrases as search terms and as the basic units in test, and their relationships with the topic of the text, has been studied in (Renouf 1993b; 1993a) and evidenced by the work in (Gay & Croft 1990) that identifies sequences of two or more nouns as an effective identification of concepts found within a document. A related area is the automatic determination of text themes, or topics that are emphasized in the text and represented by selected text excerpts. More complex methodologies such as this appear in more recent work by (Salton *et al.* 1996; 1994) that uses adjacent words to describe simple themes, and non-adjacent text possibly spanning multiple paragraphs to define more complex themes. We too use words and word-phrases to identify topics or trends in text databases.

Methodology

The methodology we describe is a general approach that can be applied to text databases of varying complexity. The results of the mining are a set of phrases that occur frequently in the underlying documents and that match a query supplied by the user. Our methodology has three major steps:

- Identifying frequent phrases using sequential patterns mining;
- Generating histories of phrases;
- Finding phrases that satisfy a specified trend.

These steps are described next.

Identifying phrases

We denote a word w by (w) and a phrase p by $< (w_1)(w_2) \ldots (w_n) >$. To capture the notion of phrases with more complex structure, we define a 1-phrase as a list of elements where each element is itself a phrase, and a *k-phrase* as an iterated list of phrases with k levels of nesting. For instance, a 1-phrase could be < <(IBM)> <(data) (mining)> >. Based on user-specified parameters, this phrase could correspond to "IBM" and "data mining" occuring in a single paragraph, with "data mining" being contiguous words in the paragraph. A 2-phrase could be < <(IBM)> <(data) (mining)> > < <(Anderson) (Consulting)> <(decision) (support)> >, with "Anderson Consulting" and "decision support" occuring in a different paragraph from "IBM" and "data mining".

We cast the phrase-identification problem as the problem of mining sequential patterns (Agrawal & Srikant 1995). The input to the latter problem is a set of sequences, called data-sequences. Each data-sequence is a list of transactions, where each transaction is a set of items (literals). For example, the following is a sequence: <(3) (4 5) (7)>, where (3), (4 5), and (7) are each transactions. Typically there is a timestamp associated with each transaction. A sequential pattern also consists of a list of sets of items; each set of items is called an element of the pattern. The *support* of a sequential pattern is the percentage of data-sequences that contain the pattern. The problem is to find all sequential patterns whose support is greater than a user-specified minimum support. This work was later extended and generalized in (Srikant & Agrawal 1996). The generalizations included adding time constraints that specify a minimum and/or maximum time period between adjacent elements in a pattern, and allowing items in an element of a sequential pattern to be present in a set of transactions whose timestamps are within a user-specified time window rather than a single transaction.

We can map a word in a text field to an item in a data-sequence or sequential pattern and a 0-phrase to a sequential pattern that has just one item in each element. Each element of a data sequence in the sequential pattern problem has some associated timestamp relative to the other elements in the sequence thereby defining an ordering of the elements of a sequence. So that we may utilize the work on sequential patterns, we treat the words as elements but redefine the timestamp for each word to a transaction ID that instead specifies both the order of occurrences of the words in the document and the locations of the words relative to grammatical sections of the document (such as sentences and paragraphs). We are then able to apply existing sequential pattern algorithms to the transaction ID labeled words to identify simple phrases from the document collection.

We allow considerable latitude in the definition of a "phrase". For instance, the user may be interested in phrases that are contained in individual sentences only. Alternatively, the words comprising a phrase may come from sequential sentences so that a phrase spans a paragraph. This generalization can be accommodated by the use of distance constraints that specify a minimum and/or maximum gap between adjacent words of a phrase. For example, the first variation described above would be constrained by specifying a minimum gap of one word and a maximum gap of one sentence. The second variation would have a minimum gap of one sentence and a maximum gap of one paragraph. For this latter example, we could further generalize the notion from a *single* word from each sentence to a *set* of words from each sentence by using a sliding transaction time window within sentences.

The generalizations made in the GSP algorithm for

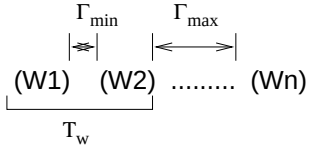

Figure 1: Minimum and maximum distance between adjacent words in a phrase

mining sequential patterns (Srikant & Agrawal 1996) allows us to do a one-to-one mapping of the minimum gap, maximum gap, and transaction window to the parameters of the algorithm. Figure 1 illustrates the affect of these parameters. In the implementation, of course, we must supply gap distances measured in discrete units rather than simply 'words', 'sentences', or 'paragraphs'. The solution is to define sufficiently large incremental values to a particular word's transaction ID when a sentence boundary is crossed and then again with a larger value when a paragraph boundary is crossed.

We extend the basic mapping of phrases to sequential patterns by providing a hierarchical mapping over sentences, paragraphs, or even sections of a text document. This extended mapping allows us to take advantage of the structure of a document to obtain a richer set of phrases. Where a document has completely separate sections, we can now mine phrases that span multiple sections thereby discovering a new set of relationships. Consider for example the base case of identifying phrases where the words are all sequential and are contained in the same sentence. Now, we may be interested in identifying phrases of the base case phrases. Termed 1-phrases, these "lists-of-lists" represent a departure over existing definitions of sequences or sequential patterns that typically represent lists of sets. Further, we can generalize our notion of having hierarchical mappings to that of nested lists-of-lists-of.... This enhancement of the GSP algorithm can be implemented by changing the apriori-like candidate generation algorithm (Agrawal *et al.* 1996) to consider both phrases and words as individual elements when generating candidate k-phrases. The manner in which these candidates are counted would similarly change.

Mining over structured data (that is, documents that have several distinct attributes each containing text) provides additional problems than simple free text, but is similar to our approach in handling sentence and paragraph boundaries. By simply treating different text attributes as different sections of the same document we are able to mine across fields just as easily as we would within one field.

Generating a history of phrases

We partition documents based upon their timestamp. The granularity of the partitioning is specified by the user. For example, partitioning documents by year may be appropriate for patent data whereas partitioning by month may be more suitable for internet-related documents. For each partition we generate a set of frequent phrases using the mappings defined earlier. This results in a history of support values for each phrase. When a particular phrase is not supported in a given partition its history will be empty for that time period. Next, we describe how to identify trends using these histories.

Identifying trends

By maintaining a support history for each supported k-phrase we can query the set of histories to select those phrases that have some specific shape in their histories. We propose the use of a shape definition language called $\mathcal{SDL}$ (Agrawal *et al.* 1995) to define the users' queries and retrieve the associated objects. There are several benefits for using a shape query language such as $\mathcal{SDL}$ to identify trends: First, the language is small, yet powerful, allowing a rich combination of operators. Second, it is a fairly straightforward task to rewrite a shape the user may define graphically, as is done in our PatentMiner system described in Section 4, into the $\mathcal{SDL}$ set of operators. Third, $\mathcal{SDL}$ allows a "blurry" match where the user may care about the overall shape but does not care about specific details of each interval of the shape. Finally, $\mathcal{SDL}$ allows itself to be implemented efficiently since most of the operators are designed to be greedy to reduce non-determinism which in turn reduces the amount of back-tracking that must be done across the histories.

Trends are simply those k-phrases selected by the shape query with the additional information of the time periods in which the trend is supported.

Experience

Figure 2 shows a high-level view of our system to compute and visualize the word-phrase trends, which we now describe.

The PatentMiner System

The PatentMiner prototype is a system we developed to discover trends among patents granted in different categories. The system is connected to an IBM DB2 database containing all granted U.S. Patents and patent data is retrieved using a dynamically generated SQL query based upon the selection criteria specified by the user. The system allows selection of patents in a specific classification or by keywords appearing in the title or abstract of the patents. Once retrieved, a histogram displaying the number of patents for each year is shown and the user may then specify a range of years upon which the system will focus.

Next, the user can choose the maximum and minimum gap desired between words in the phrases to be mined, as well as the minimum support all phrases must meet for each time period between the start and ending years. Finally, a shape matching the desired trend (such as "recent upwards trend", "recent spike

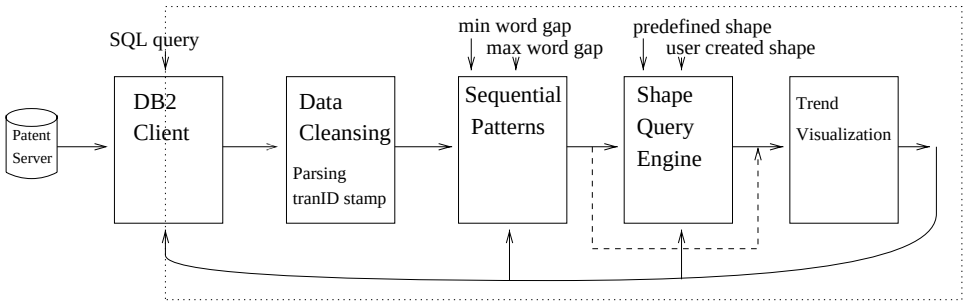

Figure 2: The PatentMiner system

in usage", "downwards trend", and "resurgence of usage") is selected and the mining process begins. Alternatively, users can define their own shape by using a visual shape editor. Once the phrases matching the shape query are found, they are presented in a visual display.

We now describe the internal processes of the system. The text patent data is first cleansed to remove stopwords, while at the same time assigning transaction IDs to the words depending on their placement within the patent. The transaction IDs encode both the position of each word within the document as well as representing sentence, paragraph, and section breaks, and are currently represented as long integers with the sentence boundaries using the 1,000th location, the paragraph boundaries using the 100,000th location, and the section boundaries using the 10,000,000th location. By specifying a minimum gap of 1,000, for instance, phrases would consist of words each from different but sequential sentences. Further, as the data is parsed it is partitioned based on the year the particular patent was granted.

For each partition of cleansed patent data, we perform a mining pass using the Generalized Sequential Patterns (GSP) algorithm (Srikant & Agrawal 1996) to generate those phrases in each partition that meet the minimumsupport threshold. The resulting phrases are then saved so that many different shape queries can be run against them. The Shape Query Engine takes the set of partitioned phrases for the years of interest and selects those that match the given shape query. Once a shape query has been defined, either internally or using the graphical editor, a rewriting of the query into $\mathcal{SDL}$ (Agrawal *et al.* 1995) is performed. Given the shape query in Figure 3, the rewriting of this query into $\mathcal{SDL}$ is shown in Figure 4. The rewriting happens as follows. For every partitioned time period of documents there is a corresponding interval in the shape query graph that has associated with it beginning and ending relative levels of support. In the case where every interval has a specific beginning or ending value, the rewriting into SDL is straightforward in that the slope of each interval determines the basic shape query that is used for that interval. For example, intervals with a positive

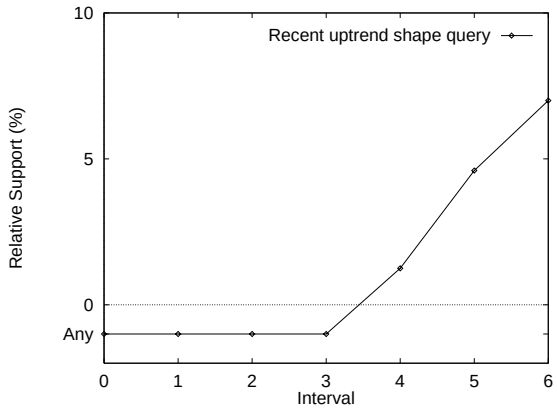

Figure 3: A recent Uptrend shape query

```
(shape strongUp ( ) (comp Bigup Bigup Bigup ))
(files "list_hist" )
(query (window 1) ((strongUp) (support 2 end)))
(quit)
```

Figure 4: Recent uptrend query

slope translate to an "up" shape of length one, while intervals with a negative slope translate to a "down" shape of length one. The concatenation of all of these base shapes then defines our SDL query. In the case where only some of the intervals of a shape query have been specified, as in intervals three to six in Figure 3, then the same concatenation occurs but the resulting SDL shape can have any shape match the unspecified intervals.

Trends from the Patent Database

We present some of the trends our system found from U.S. Patents classified in the category "Induced Nuclear Reactions: Processes, Systems, and Elements" in Figure 5. These example phrases matched a shape query that represented an increasing trend of their usage in recent years. Without knowing a priori the kind of patents filed in this category, we are able to look at the trends and determine some of the popular topics of recently granted patents.

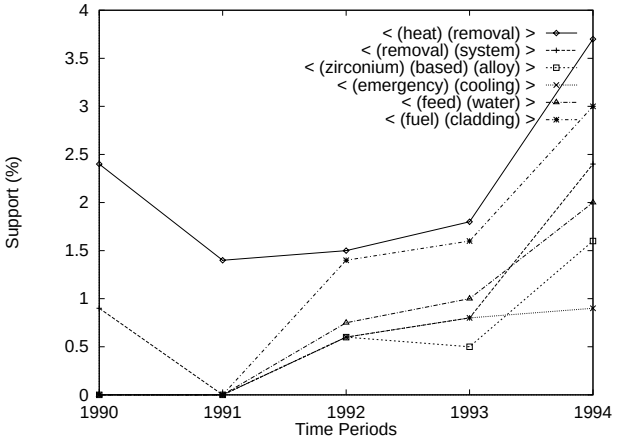

Figure 5: Some recent upwards trends

A potential problem with this system is that the number of phrases that match a query can be quite large. There are two types of pruning we use to reduce the number of phrases to a more reasonable number. The first form of pruning is to drop non-maximal phrases when their support is near that of a maximal phrase that is a superset. The second form of pruning involves the use of a syntactic hierarchical ordering of phrases. The intuition is that if phrase X is a syntactic sub-phrase of phrase Y, then the concept corresponding to X is usually a generalization of the concept corresponding to Y. Users initially see only the most general concepts, and can explore lower-level concepts by selecting some of the phrases.

Conclusion

In this paper we have investigated the application of existing data mining algorithms to identify trends in large text databases. We have shown how the phrase-identification problem can be casted as a sequential patterns problem and how to reuse existing data mining algorithms to mine text. The use of a shape-based query language for identifying trends over the mined text data was also described. We built a working text mining system, called PatentMiner, to demonstrate the usefulness of our approach. Scaleup experiments show that PatentMiner scales approximately linearly with the number of patents in the database.

Acknowledgments We are grateful to the IBM Almaden Patent Server team, especially Laura Anderson, Steve Boyer and Tom Griffin for their ongoing contributions and suggestions. Thanks also to Peter Schwartz, Laura Haas, and Guy Lohman for their help with database inter-operability.

References

Agrawal, R., and Srikant, R. 1995. Mining sequential patterns. In *Proceedings of the 11th International Conference on Data Engineering.*

Agrawal, R.; Psaila, G.; Wimmers, E.; and Zait, M. 1995. Querying shapes of histories. In *Proceedings of the 21st International Conference on Very Large Databases.*

Agrawal, R.; Mannila, H.; Srikant, R.; Toivonen, H.; and Verkamo, A. I. 1996. Fast Discovery of Association Rules. In Fayyad, U. M.; Piatetsky-Shapiro, G.; Smyth, P.; and Uthurusamy, R., eds., *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press. chapter 12, 307–328.

Croft, W.; Turtle, H.; and Lewis, D. 1991. The use of phrases and structured queries in information retrieval. In *14th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 32–45.

Deerwester, S.; Dumais, S. T.; Furnas, G. W.; Landauer, T. K.; and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41(6):391–407.

Feldman, R., and Dagan, I. 1995. Knowledge discovery in textual databases (KDT). In *Proceedings of the 1st International Conference on Knowledge Discovery in Databases and Data Mining.*

Feldman, R., and Hirsh, H. 1996. Mining associations in text in the presence of background knowledge. In *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases and Data Mining.*

Gay, L., and Croft, W. 1990. Interpreting nominal compounds for information retrieval. *Information Processing and Management* 26(1):21–38.

Lewis, D., and Croft, W. 1990. Term clustering of syntactic phrases. In *13th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 385–404.

Renouf, A. 1993a. Making sense of text: automated approaches to meaning extraction. *17th International Online Information Meeting Proceedings* 77–86.

Renouf, A. 1993b. What the linguist has to say to the information scientist. *Journal of Document and Text Management* 1(2):173–190.

Salton, G.; Allan, J.; Buckley, C.; and Singhal, A. 1994. Automatic analysis, theme generation, and summarization of machine readable texts. *SCIENCE* 264(5164):1421–1426.

Salton, G.; Singhal, A.; Buckley, C.; and Mitra, M. 1996. Automatic text decomposition using text segments and text themes. In *Proceedings of Hypertext*, 53–65.

Srikant, R., and Agrawal, R. 1996. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the Fifth International Conference on Extending Database Technology (EDBT).*