JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 10, Number 6, 2003

Pp. 821–855

Mining the Biomedical Literature in the Genomic Era: An Overview

HAGIT SHATKAY[1] and RONEN FELDMAN[2]

ABSTRACT

The past decade has seen a tremendous growth in the amount of experimental and computational biomedical data, specifically in the areas of genomics and proteomics. This growth is accompanied by an accelerated increase in the number of biomedical publications discussing the findings. In the last few years, there has been a lot of interest within the scientific community in literature-mining tools to help sort through this abundance of literature and find the nuggets of information most relevant and useful for specific analysis tasks. This paper provides a road map to the various literature-mining methods, both in general and within bioinformatics. It surveys the disciplines involved in unstructured-text analysis, categorizes current work in biomedical literature mining with respect to these disciplines, and provides examples of text analysis methods applied towards meeting some of the current challenges in bioinformatics.

Key words: biomedical literature mining, information retrieval, information extraction, text mining, PubMed, genomics.

1. INTRODUCTION

THE LAST DECADE HAS BEEN MARKED by unprecedented growth in both the production of biomedical data and the amount of published literature discussing it. Advances in computational and biological methods have remarkably changed the scale of biomedical research. Complete genomes can now be sequenced within months and even weeks (Myers, 1999; Venter *et al.*, 2001), computational methods expedite the identification of tens of thousands of genes within the sequenced DNA (Bafna and Huson, 2000; Burge and Karlin, 1998; Korf *et al.*, 2001), and automated tools are developed for analyzing properties of genes and proteins (Altschul *et al.*, 1997; Emanuelsson *et al.*, 2000; Horton and Nakai, 1997; Jaakkola *et al.*, 2000; Sonnhammer *et al.*, 1998). Modern techniques such as DNA microarrays allow simultaneous measurements for all genes/proteins expressed in a living system (DeRisi *et al.*, 1997; Lockhart *et al.*, 1996; Schena *et al.*, 1995; Spellman *et al.*, 1998). These large-scale experimental methods produce large quantities of *data.* When processed, the data can provide actual *information* about gene expression patterns, for instance, which genes are expressed in various tissues, and which ones are over/under expressed at the onset of a disease or during a specific phase of the cell development. Still, the ultimate goal of conducting large-scale biology (Bassett *et al.*, 1999) is to translate these large amounts of *information* into *knowledge*

[1]School of Computing, Queen's University, Kingston, Ontario, Canada K7L3N6.
[2]Department of Computer Science, Bar Ilan University, Ramat Gan 52900, Israel.

of the complex biological processes governing the human body and to utilize this knowledge to advance healthcare and medicine.

Almost every known or postulated piece of information pertaining to genes, proteins, and their role in biological processes is reported somewhere in the vast amount of published biomedical literature. However, the advancement of genome sequencing techniques is accompanied by an overwhelming increase in the literature discussing the discovered genes. This combined abundance of genes and literature produces a major bottleneck for interpreting and planning genome-wide experiments. Thus, the ability to rapidly survey this literature constitutes a necessary step toward both the design and the interpretation of any large-scale experiment. Moreover, automated literature mining offers a yet untapped opportunity to integrate many fragments of information gathered by researchers from multiple fields of expertise into a complete picture exposing the interrelated roles of various genes, proteins, and chemical reactions in cells and organisms.

During the last few years, there was a surge of interest in using the biomedical literature, (e.g., Andrade and Valencia, 1997; Craven and Kumlien, 1999; Friedman *et al.*, 2001; Fukuda *et al.*, 1998; Hanisch *et al.*, 2003; Jenssen *et al.*, 2001; Leek, 1997; Rindflesch *et al.*, 2000; Shatkay *et al.*, 2000; Yandell and Majoros, 2002), ranging from relatively modest tasks such as finding reported gene location on chromosomes (Leek, 1997) to more ambitious attempts to construct putative gene networks based on gene-name co-occurrence within articles (Jenssen *et al.*, 2001). Since the literature covers all aspects of biology, chemistry, and medicine, there is almost no limit to the types of information that may be recovered through careful and exhaustive mining. Some possible applications for such efforts include the reconstruction and prediction of pathways, establishing connections between genes and disease, finding the relationships between genes and specific biological functions, and much more. It is important to note that a single mining strategy is unlikely to address this wide spectrum of goals and needs.

Regardless of the explicit goal, there are several major hurdles to overcome when using the biomedical literature for finding information. The most obvious is the sheer number of available articles, which is continuously growing. For instance, the most widely used biomedical literature database, NCBI's PubMed, contains over 12,000,000 abstracts. A query for abstracts mentioning *gene* or *protein* returns about 3,000,000 articles, of which nearly two thirds were published just within the past decade. We note that this prolific database by no means covers *all* the publications in all the areas related to biomedicine, but rather, just those meeting certain criteria.

Another major problem arises when searching for the literature relevant to specific entities such as a gene, a protein, or a disease. Since both the English language and the biomedical jargon suffer from several levels of ambiguity, we may miss relevant papers, as well as retrieve irrelevant ones. Both of these issues are discussed in more detail in the next section.

Yet another issue is the inherent difference between the text that is typically searched by current text-handling tools and the scientific literature. Much of the work on text mining is aimed at and tested on articles such as news reports, typically written by professional writers whose main goal is to clearly convey a story to the average reader. In contrast, scientific documents are written by scientists whose first language may often not be English, whose main focus is research rather than report writing, and whose target audience is a relatively small group of fellow scientists, all familiar with the same domain-specific jargon. Scientific articles thus often use nonstandard terms and grammatical structures and include material and background information that may not directly pertain to—or may even *contradict*—the paper's main point. All these factors add a level of complexity to the scientific literature, making it harder to mine with standard tools.

The automated handling of text is an active research area, spanning several disciplines. These include the following: *information retrieval*, which mostly deals with finding documents that satisfy a particular information need within a large database of documents (for an introduction see, for instance, Sahami [1998], Salton [1989], Witten *et al.* [1999]); *natural language processing (NLP)*, a broad discipline concerned with all aspects of automatically processing both written and spoken language (Allen, [1995], Charniak [1993], and Russell and Norvig [1995] are some introductory references); *information extraction (IE)*, a subfield of NLP, centered around finding explicit entities and facts in unstructured text (e.g., Cardie, 1997; Cowie and Lehnert, 1996). For instance, identifying all the positions in the text that mention a protein or a kinase (entity extraction), or finding all phosphorylation relationships to populate a table of phospohrylated proteins along with the responsible kinase (relationship extraction) are both IE tasks. Finally, *text mining*

(Hearst, 1999), the combined, automated process of analyzing unstructured, natural language text in order to discover information and knowledge that are typically difficult to retrieve.

In this paper, we provide a self-contained introduction to the above methods and survey the specific work done on literature mining for bioinformatics, placing the latter within the context of the general methods. We hope that this presentation of the current state of the art in biomedical literature mining will help set the stage for future productive work in this emerging field. As a lot of recent work has been published in this domain, we concentrate on representative examples using each technique, rather than being all-inclusive.

The next section presents the various methods in text mining, including retrieval, extraction, and NLP, followed by a survey of the work on literature mining in the context of bioinformatics. We then provide two examples, drawing on our experience of developing an information retrieval and an information extraction system for finding information about genes. The first is GenTheme, a system for finding functional relations among genes based on PubMed abstracts (Shatkay *et al.*, 2000). The second, LitMiner, is an information extraction system that was used for finding information about gene expression and protein products within full-text articles. The latter system was the winning entry in the information extraction task of the 2002 KDD cup (Regev *et al.*, 2002; Yeh *et al.*, 2002).

2. METHODS IN TEXT PROCESSING

This section introduces the disciplines involved in text processing along with the techniques and methods they use. We start with general techniques from natural language processing and text mining. We then proceed to the more specific areas of information retrieval and information extraction. The former is centered around the high-level task of identifying relevant documents that satisfy a particular information need, without much concern to issues of natural language representation/understanding. On the other hand, information extraction handles the extraction of specific entities, facts, and events from within the text and is anchored in NLP techniques. We conclude the section with a short review of the standard evaluation methods employed in these fields.

2.1. Natural language processing: General techniques

Natural language processing is concerned with all aspects and stages of converting spoken, handwritten, or printed text from raw signal to information that can be used by either humans or automated agents. In the context of bioinformatics, we are concerned only with printed text that is already stored in a machine-accessible format and therefore concentrate on common text processing operations (Allen, 1995) as used by typical text mining systems. These include the tokenization and zoning tasks, part of speech tagging, and (shallow) parsing.

2.1.1. Tokenization. The first step in text analysis is the process of breaking the text up into its constituent units—or its *tokens*. This process is known as *tokenization*. Tokens may vary in granularity depending on the particular application. Consequently, tokenization can occur at a number of different levels: the text could be broken up into chapters, sections, paragraphs, sentences, words, syllables, or phonemes. For any level of tokenization, many different algorithms exist for breaking up the text. The most common form of tokenization in mining systems is the fragmentation of text into words and sentences. The main challenge of fragmentation at the sentence boundaries is distinguishing between a period that signals an end of sentence and a period that is part of a previous token like the shorthand *Mr.*, *Dr.*, etc.

2.1.2. Part of speech tagging. Part-of-speech tags are a set of word-categories based on the role that words may play in the sentence in which they appear. *Part of Speech (PoS) Tagging* is the annotation of words with the appropriate PoS tags, based on their context within the sentence. PoS tags convey information about the semantic content of a word. *Nouns* usually denote tangible and intangible entities while *prepositions* express relationships between entities. While sets of tags may vary, most part-of-speech tag sets make use of the same basic categories. The most common set contains seven different tags: *Article, Noun, Verb, Adjective, Preposition, Number,* and *Proper Noun*. Some systems use a much more elaborate set of tags. For example, the complete Brown Corpus (Francis and Kucera, 1979) tag-set has 87 basic tags.

Several approaches exist to PoS tagging. The most common taggers are either *rule-based* taggers or *probabilistic* ones based on hidden Markov models (HMMs). HMM-based taggers (Charniak, 1993; Dermatas and Kokkinakis, 1995; Elworthy, 1994; Kupiec, 1992; Merialdo, 1994; Weischedel *et al.*, 1993) estimate the probability of a sequence of part-of-speech tags to be assigned to a given sequence of words, based on a probabilistic (Markov) model. In order to estimate the model parameters, the tagger undergoes a training phase, using an annotated corpus, such as the WSJ corpus in the Penn Treebank (Marcus, 1992). The latter consists of about one-million tagged words. Using a tri-gram model (that is, a model in which the current word-tag depends only on the tags assigned to the two preceding words), HMM-based taggers have achieved 94–96% accuracy on held-out test sets, i.e., sets other than the ones used for training the model.

On the other hand, typical rule-based approaches (Brill, 1992; Brill, 1999; Greene and Rubin, 1971) rely on rules that use contextual information to assign tags to unknown or ambiguous words. These rules are often known as *context frame rules.* For instance, a context frame rule might say: "*If an ambiguous/unknown word X is preceded by a determiner and followed by a noun, tag it as an adjective*."

In addition to contextual information, many rule-based taggers use morphological information to aid in the disambiguation process. For example (Maltese and Mancini, 1991), if an ambiguous/unknown word ends with an *ing* suffix and is preceded by a verb, it may be tagged as a verb. Another source of hints for the correct tagging of words can be obtained from orthography such as capitalization and punctuation. For some languages, such as English and German, information about capitalization proves extremely useful in the tagging of unknown nouns; usually capitalized nouns would be tagged as proper nouns. In other languages, such as Hebrew and Arabic, there are no capital letters; hence, no hints can be derived from orthography.

Initially, rule-based taggers required human-tagged training sets, for what is known as *supervised* learning of rules. However, more recently, several researchers (Hindle, 1989; Schütze, 1993) started to work on *unsupervised* rule-learning, or *bootstrapping*. Starting with an untagged text corpus and a coarse, generic tagger, the tagger assigns tags to the corpus. An expert reviews the tagged text and corrects any mistakes found. In practice, the expert does not typically have to correct more than 20% of the words. The corrected tagging is then run again through the tagger, where special emphasis is placed on words which were erroneously tagged in the first phase. This iterative process, of expert review followed by a tagger rerun, may be repeated until an acceptable error rate is reached.

2.1.3. Parsing and shallow parsing. Parsing is the process of determining the complete syntactic structure of a sentence or a string of symbols in a language. A parser usually takes as its input a sequence of tokens that were extracted from the original text by a lexical analyzer. The output from the parser is typically an abstract syntax tree, whose leafs correspond to the individual words (lexemes) in the text, and whose internal nodes represent syntactic structures, identified by grammatical tags, such as *Noun, Verb, Noun Phrase, Verb Phrase*, etc. Efficient and accurate parsing of unrestricted text is not within the reach of current techniques. Standard algorithms are too expensive to use on very large corpora and are not robust enough.

A practical alternative is *shallow parsing*. This is a coarser process of breaking documents into nonoverlapping word sequences or *phrases*, such that syntactically related words are grouped together. Each phrase is then tagged by one of a set of predefined grammatical tags such as *Noun Phrase, Verb Phrase, Prepositional Phrase, Adverb Phrase, Subordinated clause, Adjective Phrase, Conjunction Phrase,* and *List Marker*. Shallow parsing has the benefit of both speed and robustness of processing, which comes at the cost of compromising the depth and fine-granularity of the analysis. Shallow parsing is generally useful as a preprocessing step, either for bootstrapping—extracting information from corpora for use by more sophisticated parsers—or for end-user applications such as information extraction. Shallow parsing allows the identification of relationships between the object, the subject, and any other spatial or temporal phrases within a sentence.

2.2. Text mining

Text mining is the combined, automated process of analyzing unstructured natural language text in order to discover information and knowledge that are typically difficult to retrieve (Hearst, 1999). It uses techniques from the general field of data mining (Frawley *et al.*, 1991), but since it handles unstructured data, a major part of the process deals with the crucial stage of pre-processing the document collections

(using techniques such as text categorization [discussed in Section 2.3.4], term extraction [Daille *et al.*, 1994; Frantzi, 1997], and information extraction [discussed in Section 2.4]). In addition to preprocessing the documents, text mining also covers the storage of the intermediate representations, the techniques to analyze these intermediate representations, such as distribution analysis, clustering (e.g., Sahami *et al.*, 1996; Goldszmidt and Sahami, 1998), trend analysis (Lent *et al.*, 1997), association rules (e.g., Rajman and Besançon, 1997), and the visualization of the results (Aumann et al., 1999; Stapley and Benoit, 2000).

A typical text mining system starts with collections of raw documents, without any labels or tags. Documents are first automatically tagged by categories, or by terms or relationships extracted directly from the documents. This process is called *text categorization*, and it partitions a large collection of documents into subsets that are interrelated by some predefined criteria. As this is a subarea of information retrieval, we review it in more detail in Section 2.3.4

In the following phase, the assigned categories, along with entities and relationships found in the text, are used to support data mining operations on the documents. Limiting the set of mined documents to certain relevant subcategories somewhat simplifies the follow-up tasks and increases the likelihood that the mining tools applied to the subcategories will extract the most on-target information from the text. The actual detection of facts within the text is typically performed through information extraction methods, as discussed in Section 2.4.

2.3. Information retrieval

Information retrieval is concerned with identifying documents that are most relevant to a user's need within a very large set of documents. More precisely, given a large database of documents, and a specific information need—usually expressed as a *query* by the user—the goal of information retrieval methods is to find the documents in the database that satisfy the information need. Naturally, the task has to be performed accurately and efficiently.

2.3.1. Boolean queries and index structures. There are several ways to express, as well as to satisfy, the information need. A simple and common way for a user to express her need is through a *Boolean* query. Under this setting, the user provides a term (e.g., *OLE1*), or a Boolean term-combination (e.g., *OLE1* and *lipid*). The result is the set of *all* the documents in the database satisfying the query constraints, e.g., containing both the query terms *OLE1* and *lipid*. This query paradigm is used by the biomedical literature database PubMed and by many other text databases and search engines over the world wide web. It is supported by an index covering all the terms in the whole database of documents. Each *term* may be a single word (e.g., *blood*) or a phrase (e.g., *blood pressure*). It is common practice to omit from the index terms that are frequent and non–content–bearing, such as prepositions. These terms are usually referred to as *stop words* and are viewed as delimiters when processing text. The index structure contains all the terms, typically sorted alphabetically for quick access, and holds for each term a reference to all the documents in the database that contain it, as demonstrated in Fig. 1.

When a user poses a query, the index structure is efficiently searched for the query terms occurring in it, and all the documents found to contain the terms (or the boolean combination of the terms) are retrieved. There are various methods to create indices and use them. A full description is beyond the scope of this paper. Further information on this subject is available in books concerning databases and information access, such as the one by Witten *et al.* (1999).

The simple form of Boolean query, which has the advantage of efficient implementation over large databases, suffers several limitations:

1. The number of documents typically retrieved is *prohibitively large*.
2. A substantial part of the retrieved documents are *irrelevant* to the user's information need.
3. Many relevant documents *may not be retrieved*. For instance, if we were retrieving from PubMed, using the query *OLE1*, abstracts discussing *OLE1* under any other of its aliases (e.g., *DNA repair protein* or *fatty-acid desaturase 1*) would not be retrieved.

Problem 2 above stems from the well-known *polysemy* phenomenon: a word may have multiple meanings in different contexts. For instance, when looking in PubMed for the term *Cytosine Deaminase* under its acronym *CD*, we may retrieve all abstracts referring to *Cytosine Deaminase* in which we are actually

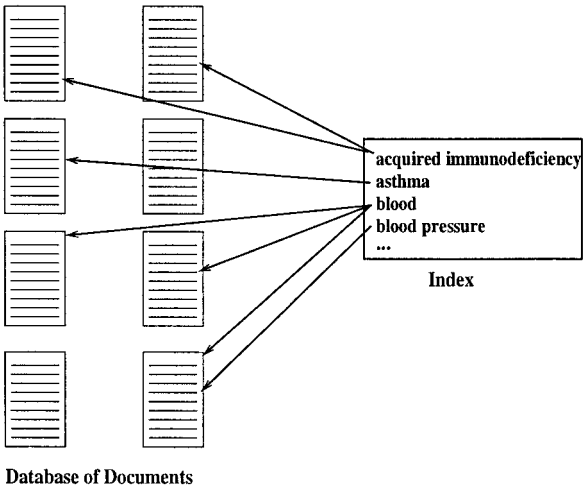

FIG. 1. An index relating terms to the documents in which they occur.

interested, but also all those discussing *Crohn's Disease* (also *CD*) which are completely unrelated. On the other hand, limitation 3, stems from *synonymy*: a single concept is discussed in various abstracts under different names.

2.3.2. Similarity queries and the vector model. A broadly used alternative to the Boolean query is the *similarity query*, which is typically based on the *vector space* model, discussed throughout this section.

Under this setting, documents are viewed as vectors over terms, as we formally define below. A query, q, which may consist of many terms, and may even comprise a complete document, is in-and-of-itself viewed as a body of text, rather than merely as a search-terms combination. Thus, it too is represented as a vector. The retrieval task reduces to searching the database for document vectors that are *most similar* to the query vector. Various similarity measures over documents have been devised and used (Salton, 1989; Goldszmidt and Sahami, 1998).

To explicitly define the vector model, we refer to the large set of documents from which retrieval is conducted as the *database* and denote it as *DB*. The *vocabulary* of the database is the set of all the terms occurring within *DB*'s documents. Let M be the number of distinct terms $\{t_1, \ldots, t_M\}$ in this vocabulary. As stated before, a term, t_i, may be a single word or a longer phrase such as "*blood pressure*" or "*acquired immunodeficiency syndrome*." Moreover, stop-words are typically removed in a preprocess. Some systems may also *stem* words, removing common suffixes such as *ing* or *es* (see, for instance, Porter's stemming algorithm [Porter, 1997]).

A *document*, d, in the database is represented as an M-dimensional vector:

$$\langle w_{d_1}, w_{d_2}, \ldots, w_{d_M} \rangle,$$

where w_{di} is a weight representing the occurrence or the significance of the term t_i within the document. The particular choice of term-weights can significantly influence the results of a similarity search, and there are several schemes for calculating the weights.

An intuitive representation is the binary one, where the weight is either 1 or 0, corresponding to the presence or the absence of the term in the document:

$$w_{di} = \delta_{di} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } t_i \in d, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

While this representation is clear and simple, it does not account for various properties of documents and terms that may improve retrieval quality. For instance, a simple extension of the binary scheme uses the number of times the term actually occurs within the document as the weight. The intuition here is

that a document in which a query-term is over-represented has a good chance to indeed be relevant to the information need represented by the query. Formally:

$$w_{d_i} \stackrel{\text{def}}{=} n_{d_i} \text{ iff } t_i \text{ occurs } n_{d_i} \text{ times in document } d, \quad 0 \leq n_{d_i}. \tag{2}$$

The above indeed accounts for the abundance of the term in a document, but does not consider the total length of the document. A short but highly relevant document may contain fewer occurrences of its relevant terms than a long, marginally-relevant document. To correct for this, one can normalize the above weight, dividing it by the total number of term occurrences in the document, denoted by N_d. That is, rather than using n_{d_i} above, we use $\overline{n}_{d_i} \stackrel{\text{def}}{=} n_{d_i}/N_d$.

Yet another consideration is that if one query-term frequently occurs in many documents across the database, while another is rare and specialized, documents containing the rare query-term are likely to be more relevant to the user's information need than documents containing the frequent one. These intuitions are combined and formalized through a family of weighting schemes commonly known as $TF \times IDF$. The acronym stands for "Term Frequency $\times$ Inverse Document Frequency". Under this general scheme, the weight, w_{d_i}, is expressed as:

$$w_{d_i} \stackrel{\text{def}}{=} \begin{cases} r_{d_i} \cdot f_i & \text{if } t_i \text{ occurs in } d, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where r_{d_i} is a *local* measure of the occurrence of term t_i in document d, and f_i is a *global* measure, invertly proportional to the number of documents containing t_i in the whole database.

There are several ways to calculate the local measure, r_{d_i}. We have already seen some examples, including $r_{d_i} = 1$ (Eq. 1), $r_{d_i} = n_{d_i}$ (Eq. 2).[1] Other alternatives are $r_{d_i} = (1+\ln[n_{d_i}])$ or $r_{d_i} = (k+(1-k)\cdot\frac{n_{d_i}}{\max_j[n_{dj}]})$, where k is a constant, $0 \leq k \leq 1$, and the denominator is the number of occurrences of the *most frequent* term in the document d.

Similarly, there are various options for calculating the *global measure*, f_i. For example, denote by N_i the total number of documents containing the term t_i in the database, DB. A simple expression for f_i is then

$$f_i \stackrel{\text{def}}{=} \frac{1}{N_i}.$$

Other alternatives are $f_i \stackrel{\text{def}}{=} \ln[1+\frac{|DB|}{N_i}]$, where $|DB|$ denotes the number of documents in the database, or $\ln[\frac{|DB|-N_i}{N_i}]$. Further discussion of weighting schemes is available in the extensive literature on information retrieval, e.g., Salton (1989) and Witten *et al.* (1999). In a more application-specific work, Wilbur and Yang (1996) study weighting schemes pertaining to retrieval from the biomedical literature.

We have so far seen various methods used for representing documents and queries as vectors. Using this representation, we can apply vector-similarity measures and assess similarity between pairs of documents as well as between a query and each document in the database.

Many similarity measures among high-dimensional vectors exist. Perhaps the most well known, outside the information-retrieval realm, is the Euclidean distance. The smaller the distance, the more similar the vectors. Formally, the *Euclidean distance*, also known as the L_2 norm, between two n-dimensional vectors, $V_1 = \langle v_{1_1}, \ldots, v_{1_n} \rangle$ and $V_2 = \langle v_{2_1}, \ldots, v_{2_n} \rangle$, is defined as

$$d_{Euc}(V_1, V_2) = \sqrt{\sum_{i=1}^{n} (v_{1_i} - v_{2_i})^2}\,.$$

It is illustrated, for the simple two-dimensional case, in Fig. 2. As can be seen from the figure, the length of the vectors strongly effects the distance between them. In the context of documents, this typically means

[1] In both of these cases $f_i = 1$.

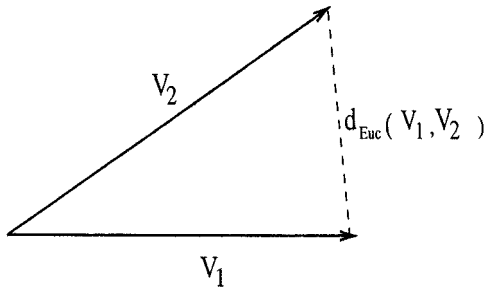

FIG. 2. The Euclidean distance (dashed) between two vectors (solid arrows).

that two documents containing many terms tend to diverge more from each other than documents containing fewer terms, regardless of the actual dissimilarity in their respective content.

A similarity measure that is widely used in information retrieval and is not as sensitive to the vector's length is the *cosine coefficient* between two vectors (see Witten *et al.* [1999] and earlier references within). This measure is the cosine of the angles between the two vectors, as illustrated in Fig. 3. Formally, the cosine coefficient between two vectors, V_1, V_2, whose respective lengths (norms) are $\|V_1\|$, $\|V_2\|$ is defined as

$$\cos(V_1, V_2) \stackrel{\text{def}}{=} \frac{\sum_{i=1}^{n} v_{1_i} \cdot v_{2_i}}{\|V_1\| \cdot \|V_2\|}.$$

Unlike the Euclidean distance, it is a *similarity*, rather than a *distance* measure. One implication is that it is not a metric; specifically, it does not satisfy the triangle inequality. Moreover, its value, which is alway in the range [0, 1], gets closer to 1 the more similar the two vectors are, and closer to 0 as the vectors diverge from one another (or get closer to being perpendicular to each other, in geometrical terms).

Under the binary representation, a simple Boolean disjunctive query (e.g., *HIV* or *AIDS*) would be a vector with 1 in the positions corresponding to the query terms and 0 everywhere else. A similarity search throughout the database for document vectors matching the query vector, using the cosine measure as defined above, would retrieve exactly the same documents as an index-based Boolean query engine. However, different weighting schemes would return different sets of documents. Moreover, for queries involving many terms, the use of similarity search has two major advantages. First, it alleviates the burden of having to specify a complicated Boolean query that may not even correctly express the information need. Second, it typically retrieves documents that fit the information need much better than any Boolean-based retrieval could and naturally ranks the retrieved documents according to the level of similarity to the set of query terms. Since queries can contain arbitrarily many terms with no explicit Boolean operators and can in fact be complete documents, it is the *word combination* that determines the result rather than any specific word.

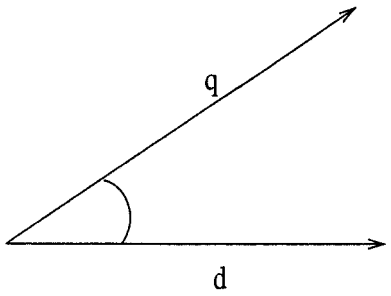

FIG. 3. The angle between two vectors, q and d.

For example, consider the query "*aids patient cancer capoci sarcoma*" against the document database PubMed.[2] By submitting this query, we are likely to find medical reports about HIV patients suffering from the Capoci sarcoma tumor, rather than about a nurse who *aids* sick *patients*. The additional words *cancer, capoci, sarcoma* in this case serve to disambiguate the context for the word *aids* to be interpreted as AIDS with relation to *HIV* and to give a higher score to documents discussing patients with HIV suffering from Capoci sarcoma than to those discussing patient assistance or people patiently waiting for an aid. The former documents simply have higher weights for most of the query terms while the latter do not. Note that no Boolean operator needs to be specified, and none of the documents returned must contain *all* of the query terms. This example shows how the polysemy in the words *aids* and *patient* is implicitly resolved by the presence of other terms, without placing a strong requirement for the occurrence of any one of the terms in the resulting documents.

2.3.3. Beyond cosine coefficient. The vector model, along with the cosine-based similarity, have proven very useful, but still have some drawbacks. We noted earlier that the prominent problems in satisfying an information need based on natural language queries are *polysemy* (multiple semantics for a single term) and *synonymy* (multiple terms with the same semantics). The vector model, together with the similarity measures, indeed addresses these problems to some extent, as demonstrated in the previous example. Still, the explicit words are present, and if all the documents in PubMed never mention *AIDS* and use only the term *HIV* to refer to patients bearing the immunodeficiency virus, we may retrieve all documents discussing Capoci sarcoma patients even if they are not infected with HIV, simply because no document matches the query term *AIDS*. Some attempts to circumvent the problem are described next.

Latent semantics indexing. A more flexible approach that depends less on the explicit query terms and, to some extent, accommodates synonyms and polysyms is *latent semantics analysis* introduced by Dumais *et al.* (Deerwester *et al.*, 1990; Dumais, 1990; Dumais *et al.*, 1988; Furnas *et al.*, 1988). Two main ideas underly this method:

- There are abstract concepts that the explicit words in the documents are trying to convey. Different word combinations may be used to identify the same concept (synonymy), while the same word may denote different concepts under different contexts (polysemy). The *semantics* of words is the concept they are conveying. While the words are overtly present in the document, the semantics is not explicitly stated and is therefore *latent*.
- A collection of documents, each represented by an M-dimensional vector, can be viewed as a matrix. As such, algebraic operators can be applied to it. One particular operator, namely *singular value decomposition*, can be used to identify and extract the "significant components" of the matrix. These are its largest k singular values, where k is much smaller than the original number of terms M. Each document in the matrix can thus be approximately represented as a linear combination of these k singular values, or equivalently, as a k-dimensional vector, rather than as the original M-dimensional vector.

By combining these two ideas, each of the k large singular values of a document collection is viewed as a surrogate for a class of terms with a common *hidden* semantics. Both queries and documents are transformed and expressed as vectors over these singular values rather than as vectors over M terms, and the similarity measure is applied to these transformed vectors, whose dimensionality is lower than that of the original term-space.

The method has shown a lot of promise, but suffers from two main drawbacks:

1. It was so far only shown effective on small collections of documents.
2. The algebraic transform to singular-values space overrides the actual words in the documents. Thus, the method does not provide the intuition or the ability to observe the terms responsible for the document similarity.

Probabilistic models. Another way to relax the dependency between retrieval results and the explicit query terms is the use of probabilistic models. Rather than require that a set (or a subset) of query terms

[2]Note that like many search engines, PubMed is not case sensitive.

occur in a document to qualify it for retrieval, the retrieval task is viewed as that of finding documents that with a *high probability* satisfy the need represented by the query.

To support this approach, a probabilistic model is devised to represent the query, the documents, and/or the user's information need. Van Rijsbergen's work is one of the earliest in this direction (van Rijsbergen, 1977). A more recent work is that by Ponte and Croft (1998) on the language-model approach. They view each document in the database as a *language model*, which roughly corresponds to a multinomial distribution over terms. A query is viewed as a sample from an unknown language model. The documents retrieved in response to a query are those most likely to be the *source language models* for the query. A related approach, probabilistic latent semantics, was introduced by Hofmann (1999), where documents are viewed as generated by a probabilistic model and the semantics of the chosen terms is stochastically determined by a set of hidden variables. Another related approach is that on probabilistic identification of *themes* by Shatkay and Wilbur (2000). It uses a view similar to that of Ponte and Croft, treating the information need as a set of Bernoulli distributions—which is the source model—and retrieving the documents that are most likely to have been generated by the source model. While this work was developed independently of Hofmann's, both use generative probabilistic models, and both use the method of expectation maximization (EM) to estimate the parameters. As the work was developed in the context of the PubMed database and was primarily applied to finding relationships among genes, we revisit it in Section 4.1.

2.3.4. Text categorization. A task often addressed by information retrieval systems is that of *text categorization*. This is the labeling of natural language texts with thematic categories from a predefined set of category tags. There are two main approaches to categorization. One is the *knowledge engineering* approach (Hayes, 1992; Hayes and Weinstein, 1990) where the user manually defines a set of rules to encode expert knowledge regarding the correct classification of documents into given categories. The other approach is based on *machine learning* (Cohen and Singer, 1999; Dumais *et al.*, 1998; Joachims, 1998; Larkey and Croft, 1996; Lewis, 1995; Lewis and Hayes, 1994; Lewis and Ringuette, 1994; Lewis *et al.*, 1996; Riloff and Lehnert, 1994; Sebastiani, 2002; Vapnik, 1995; Yang and Chute, 1994; Yang and Liu, 1999) where a general inductive process automatically builds a text classifier by training over a set of preclassified documents.

An example of the knowledge engineering approach is the CONSTRUE system (Hayes, 1992; Hayes and Weinstein, 1990) built by the Carnegie Group for Reuters. A typical rule in the CONSTRUE system consists of a condition defined as a disjunction of conjunctive clauses (a *DNF* formula) followed by the resulting category. For example, the following rule identifies articles that should be categorized as relevant to *wheat*:

```
If ((wheat & farm) or
    (wheat & commodity) or
    (bushels & export) or
    (wheat & tones) or
    (wheat & winter & soft))
then Wheat
else ~Wheat.
```

The main drawback of this approach is known as the *knowledge acquisition bottleneck.* The rules must be manually defined by a knowledge engineer interviewing a domain expert. If the set of categories is modified, these two professionals must intervene again. Hayes *et al.* (1992, 1990) reported a 90% break-even between precision and recall (see Section 2.5) on a small subset of the Reuters test collection (723 documents). However, it took a tremendous effort (several man years) to develop the system, and the test set was not significant to validate the results. It is not clear that these superb results scale up in a larger system.

The machine learning (ML) approach is based on the existence of a training set of documents, already classified into a predefined set of categories. A diagram of a typical ML-based categorization system is shown in Fig. 4. There are two main kinds of ML-based categorization, known as *hard* and *soft* classification. Under *hard* classification, every pair of a document and a category is assigned a truth value (*TRUE* if the document belongs to the category, *FALSE* otherwise). In contrast, *soft* classification entails a *ranking* by relevance of the categories for each document. Under this approach, rather than returning a truth value,

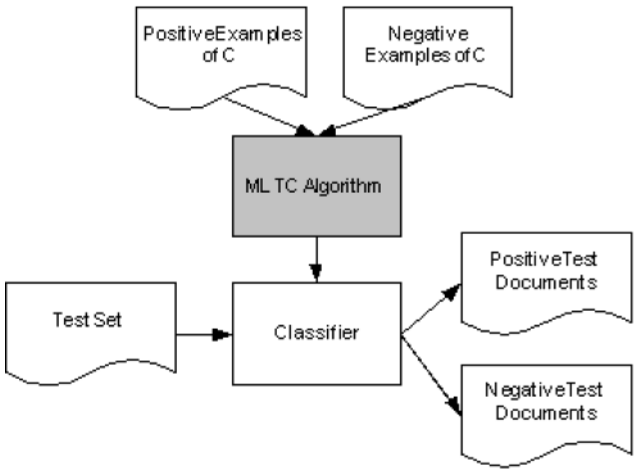

FIG. 4. Diagram of a typical ML-based categorization system.

the classifier returns a categorization status value (CSV), i.e., a number between 0 and 1 that represents the strength of evidence or the probability that the document belongs to a certain category. Documents can then be ranked with respect to each category according to their CSV value.

2.4. Information extraction

As opposed to information retrieval, which is concerned with the relatively coarse-grained task of selecting the documents most relevant to a user's need, *information extraction (IE)* is centered around the search for relevant phrases or fact-statements within an article or a text passage. Information extraction (Cardie, 1997; Cowie and Lehnert, 1996; Feldman *et al.*, 2002; Fisher *et al.*, 1995; Lehnert *et al.*, 1991) is one of the most prominent techniques currently employed in text mining. In particular, by combining NLP tools, lexical resources, and semantic constraints, it provides effective modules for mining the literature and identifying significant facts and relationships.

The extraction system looks for entities, relationships among them, or other specific facts of interest within text documents. As a first step, each document is processed to identify (extract) entities and relationships that are likely to be meaningful and content bearing. *Relationships* are facts or events involving the entities of interest. *Facts* are static in nature and usually do not change, while *events* are typically more dynamic and therefore have a specific time stamp associated with them. An example of a fact may be a statement that a gene produces a certain protein product. An event, on the other hand, is the statement that a company has entered into a joint venture to develop a new drug. The extracted information provides specific, fine-grained and well-targeted data to the mining process. Unlike the coarser tagging of documents by a fixed set of possible categories (as assigned by text categorization, see Section 2.3.4), information extraction identifies explicit concepts and relationships within the text and relates specific parts of a document to the subject of interest. It thus allows the tagging of documents by the specific entities, facts, and events found by the process, rather than by the static set of predefined categories.

2.4.1. Architecture of information extraction systems. An Information Extraction system has three to four major components. The first component is *tokenization* or *zoning*—splitting the document into basic building blocks. The typical building blocks are words, sentences, and paragraphs. In rare cases, we may choose to have higher-level building blocks such as sections or chapters. The second component is the *morphological* and *lexical* analysis—assignment of part-of-speech (PoS) tags to the words, creation of basic phrases (such as noun phrases and verb phrases), and disambiguating the sense of ambiguous words and phrases (such as polysyms). The third component is *syntactic analysis*—establishing the connection between the different parts of each sentence. This is done by performing either full or shallow parsing, as explained below. The fourth component is the *domain analysis*, where we combine all the information collected by the previous components and create complete frames that describe relationships among entities.

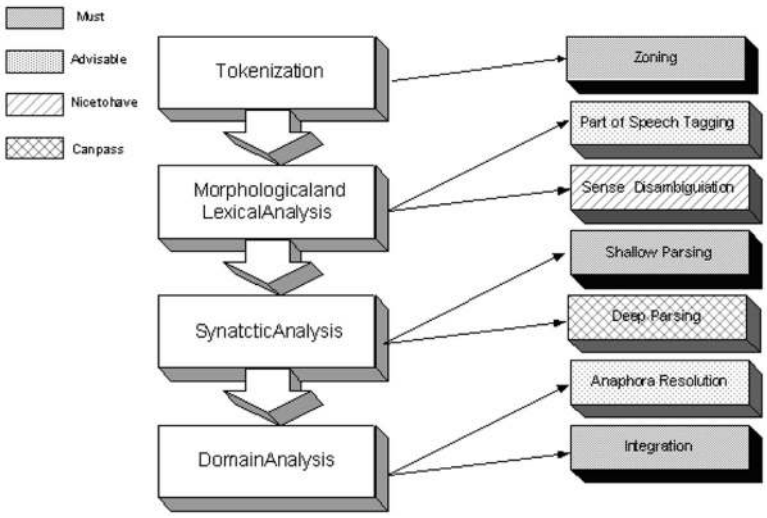

FIG. 5. Architecture of a typical information extraction system.

The domain analysis also contains an anaphora resolution component (Section 2.4.3). Figure 5 shows the architecture of a basic IE system. We elaborate on some of the subcomponents below.

2.4.2. Full versus shallow parsing in IE. We introduced the concepts of parsing and shallow parsing in Section 2.1.3. Based on actual empirical evaluation, it was found that it is enough to focus just on the core constituents of sentences and use shallow parsing augmented by *smart skips*. These skips enable the information extraction engine to skip irrelevant parts, and focus just on the important phrases of each sentence (Appelt and Israel, 1999; Feldman *et al.*, 2001; Feldman *et al.*, 2000). Researchers have attempted before to use full parsing as a component in their information systems and have concluded that it was not worthwhile to invest the extra effort. Specifically, full parsing was included in the SRI TACITUS system (Hobbs, 1991; Hobbs *et al.*, 1991) (implemented for MUC-3) and the NYU PROTEUS system (Grishman, 1995) (implemented for MUC-6). Both of these systems did not gain any improvement in accuracy due to the full parsing employed. The main problem with using full parsing is that due to the combinatorial explosion of possible parses it is both slow and very error prone.

2.4.3. Anaphora resolution. One of the main challenges in developing comprehensive text-mining systems is anaphora resolution, that is, the ability to resolve co-references (i.e., several distinct words referring to the same entity within the text) (Hobbs, 1986). Consider, for example, the following passage from a paper by Darken *et al.* (2002) about the Strabismus gene in Xenopus:

> We report here the identification of a Xenopus Stbm homolog which we have named Xstbm. This protein shows considerable homology throughout its length to Drosophila Stbm, and is nearly identical to the product of the recently cloned mouse gene Ltap (Kubar *et al.*, 2001) and to the human protein KIAA1215. The expression patterns of the Xenopus and mouse genes are also quite similar, although there are differences in detail. Both genes are expressed throughout the forming neural plate and early neural tube.

It is possible to automatically conclude that "*This protein*" at the beginning of the second sentence refers to *Xstbm*, which is the last entity of the first sentence. It is much harder to infer that the phrase "*the Xenopus and mouse genes*" in the third sentence refers to *Xstbm* and *Ltap*, respectively, since the previous sentence talked about "*This protein*" referring to *Xstbm*, but there was no mention of the word *gene*. To further infer that the term "*Both genes*" refers explicitly to *Xstbm* and *Ltap*—rather than to "*Xenopus and Mouse genes*" that were mentioned in the previous sentence—is very hard, since *Ltap* and *Xstbm* were only mentioned separately in two distinct sentences, in which *Xstbm* was referred to as a homolog and as a protein but not as a gene.

In general, it was found (Lappin and Leass, 1994) that resolving proper names and aliases (like *CD* for *Crohn's Disease*) is fairly easy, resolving pronominals such as *it, this, these, he, she*, etc. is harder, while resolving definite noun phrases such as "*the two genes*" is the most difficult and error prone. The approach often taken (Mitkov, 1998) is a knowledge-based approach where for each referring phrase all accessible antecedents are collected. The accessible antecedents are computed based on the type of the referring phrase. For proper names, all previous entities serve as candidates. For pronouns, entities that appear within the previous sentences of the current paragraph are used. For definite noun phrases, all entities that appear within the current paragraph and the preceding paragraph are used. One exception to this heuristic are entities of the form "*the X*," where *X* is a name of a company, organization, corporation, etc., and the scope is extended to the whole preceding text. In order to select the right antecedent from the set of all possible candidates, the candidates that are incompatible with the referring phrase are eliminated (either due to gender, type, or plurality). From the filtered set, the final candidate is selected according to the following heuristics, in order of importance:

1. Prefer the candidate that appears earlier in the current sentence.
2. Prefer the candidate that appears earlier in the previous sentence.
3. Prefer the candidate that appears later within other sentences (prioritized in descending order of their position in the document).

To compute the effectiveness of these anaphora resolution heuristics, the number of pairs (of referring expression and antecedent) that are correctly matched is computed. We have found that in 82% of the cases the referring expression was correctly matched to its antecedent.

2.5. Evaluation methods

When applying any form of text analysis to a body of text and, more importantly, when developing a text-analysis tool, it is critical to know how reliable the results are likely to be. While we can never anticipate all the articles we may encounter, and therefore cannot predict the performance quality in all cases, it is still useful to be able to evaluate the potential merit of a text-analysis tool by comparing its performance to that of other candidate techniques with respect to some "gold standard." Such an evaluation requires two components:

- A corpus of annotated, tagged or categorized text-items that constitute the gold standard of desired results.
- A metric or a measure that denotes how well the text-analysis system performs with respect to the gold standard.

A common way to address the second item and evaluate performance of both information extraction and information retrieval systems is by measuring *recall* and *precision* (see, for instance, Witten *et al.* [1999]). There is a set of N items, either terms, sentences, or documents, that the system essentially needs to label as "positive" or as "negative" according to some criterion—be it relevance to a specific query, membership in a document category or in a term class. Such labeling, being imperfect, partitions the original set into four subsets:

True Positives: A items *correctly* labeled as positive;
False Positives: B items *incorrectly* labeled as positive;
True Negatives: C items *correctly* labeled as negative;
False Negatives: D items *incorrectly* labeled as negative;

where the total number of items in the set is $N = A + B + C + D$.

The *Precision, P,* is the proportion of true positives with respect to all items that the system labeled as positive, while the *Recall, R*, is the proportion of true positives with respect to all items that *should be* labeled positive; that is:

$$P = \frac{A}{A + B} \quad \text{and} \quad R = \frac{A}{A + D}. \tag{4}$$

For example, suppose we are given a set of 50 documents and wish a candidate system to label them *True* if they discuss gene expression analysis and *False* otherwise. Also suppose that 30 out of the 50 documents indeed discuss gene expression analysis, while our system marks a total of 40 documents as *True*, of which only 25 truly discuss gene expression. Then the *precision, P*, achieved by our system on this set of documents is 25/40 (since only 25 out of 40 positively marked are true-positives), while the *recall, R* is 25/30 (since 25 out of the total of 30 relevant documents were recovered by the system).

A measure that combines both precision, P, and recall, R, is the *F-score*, which in its simple form (van Rijsbergen, 1979) is expressed as

$$F = \frac{2PR}{P+R}.$$

F is a number between 0 and 1, and it reaches 1 if and only if the system produces neither false-positives nor false-negatives. A more general form of the F-score allows the assignment of a higher weight to either precision or recall (Shaw *et al.*, 1997; Yang, 1999) and is calculated as

$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P + R}.$$

When $\beta = 1$, the same weight is given to precision as to recall, and F_β reduces to F.

Yet another measure of performance evaluates the total *accuracy* of the system (Yang, 1999), that is, the ratio of correct answers with respect to the total number of answers. Using the same notation as in Equation 4, the accuracy of the system is calculated as $acc = \frac{A+C}{A+B+C+D} = \frac{A+C}{N}$.

When dealing with *ranked retrieval* (e.g., using similarity measures as described in Section 2.3.2), one can look at a limited number of top-ranking documents and ask what are the precision and recall levels among those documents only. Obviously, if we were looking at the whole list (down to the lowest ranking documents), the recall will be very high, while the precision low. If we were to look at only the very top ranking documents, the recall is likely to be low, while the precision high. To account for this dependency of recall and precision on the number of ranks examined, it is common to draw a recall–precision curve with one recall–precision point at each rank. Different retrieval methods are then compared based on examination of complete recall–precision curves. Comparing curves can be complicated since some methods may seem superior to others at some retrieval rank but inferior when other ranks are considered. Some other measures of retrieval comparisons have been suggested to address this difficulty, and the reader is referred to a paper by Wilbur (1992) for a thorough discussion of these issues.

Obviously, a fair comparison of competing tools or methods requires an agreed-upon corpus of reference, reflecting some true domain, with respect to which performance is measured. Several standard text collections, as well as standardized retrieval/extraction tasks, have been devised—mostly during the past decade—especially for supporting development and evaluation of text-processing systems.

One example of a text collection is the Reuters set of articles classified into thematic categories (Lewis, 1997). It is widely used for evaluation in text-categorization research. Another text collection is the Brown Corpus (Francis and Kucera, 1979) of samples from English-American writings, categorized by types of literature (e.g., press editorials, religious writing, mystery, etc.). The corpus has a tagged version in which each word is tagged by additional information such as part of speech (noun, verb, etc.), its function (determiner, preposition), and so on. The corpus is widely used for training and testing natural language processing modules.

A forum for standardized evaluation of retrieval engines is *TREC*, the Text Retrieval Conference (Voorhees and Harman, 1993), sponsored by the National Institutes of Standards and Technology (NIST) and by DARPA. It was formed in 1992 to support large-scale evaluation of retrieval systems. Each year it offers several tracks, each of which specifies data sets and tasks to be performed on them. Participants in TREC perform the task using their respective systems and submit the results for evaluation by the NIST judges. Tracks vary by the kind of specified tasks, ranging from the video track, which is concerned with content-based retrieval from digital video, to the cross-language track, which investigates the ability of retrieval systems to identify relevant text documents for designated topics regardless of the language in which the documents are written. Very recently, a new track, *TREC Genomics*, concerned with the retrieval of genomic data from the literature and from other sources, has been formed (Hersh *et al.*, 2003).

An evaluation forum similar to TREC, but focused on *information extraction* is *MUC*, the Message Understanding Conference (NIST, 1987), which was active during the past decade. Participants in MUC tested the ability of their systems to identify entities in text (entity extraction), resolve co-reference (anaphora resolution), extract and populate attributes of entities (e.g., identify companies and extract their names, their CEO name, and their stock price, from news report text), and perform various other extraction tasks from written text.

These are a few examples of the methods and test sets used for evaluating retrieval and extraction systems. A challenge we currently face in biomedical literature mining is the creation of benchmarks and critical evaluation methods for systems developed in this very active field. The GENIA corpus (Kim *et al.*, 2003) is a collection currently being built to support training and testing of NLP methods for biomedical text mining. It consists of a few thousands PubMed abstracts annotated with biological terms. TREC genomics, which was mentioned earlier, as well as the KDD-cup challenge discussed in Section 4.2, are examples of evolving standard evaluation environments in the biomedical context. A more recent effort under way is the BioCreAtIvE, which is meant to become a standard evaluation challenge for critical assessment of information extraction systems in biology (Blaschke *et al.*, 2003).

3. TEXT MINING IN BIOINFORMATICS

The sequencing of the human genome marked the beginning of the era of large-scale genomics and proteomics. Large-scale experiments involving thousands of genes and proteins can be conducted. However, their interpretation remains a critical problem. For instance, much of the large-scale analysis of genomic data to date focuses on gene expression patterns, and particularly on establishing gene clusters based on their expression (Ben-Dor *et al.*, 1999; Sharan and Shamir, 2000; Spellman *et al.*, 1998; Tamayo *et al.*, 1999). While such methods indeed provide insight into expression correlation and other relationships among genes, this approach has clear limitations as a stand-alone analysis tool (Shatkay *et al.*, 2000). First, functionally related genes may play antagonistic roles within a pathway and thus demonstrate strong anti-correlation in their expression levels. On the other hand, genes sharing similar expression profiles may be involved in distinct biological processes. Moreover, genes may play multiple roles in complex, interrelated biological processes and share commonalities with genes in more than a single cluster. Most importantly, even when similar expression levels indeed correspond to strongly related functions, the functional relationships among genes cannot be determined from the cluster data alone. Explaining the formed gene clusters requires a lot of further analysis.

The information needed for such analysis can often be found in the published literature. However, the conventional method for finding it has been for individuals to search through the literature, gene by gene, or rely on their own knowledge of the biological process. While this procedure can be effective on a very small scale, it does not scale up well to accommodate the simultaneous analysis of thousands of genes, the relationships among them, and their role in biochemical reactions.

The prevailing on-line source for biomedical abstracts is the PubMed[3] database, which is maintained by the National Center of Biotechnology Information (NCBI). It contains over 12,000,000 scientific abstracts and is accessed by millions of users from all over the world on a daily basis. A typical search for relevant literature within PubMed starts with a Boolean query; the user provides a term or a Boolean term combination (e.g., *OLE1* and *lipid*). The result is the set of *all* the abstracts in *PubMed* satisfying the query constraints, as discussed in Section 2.3.1. We note that the lack of uniformity in nomenclature used by authors aggravates the problem of synonymy. For instance, a search for abstracts about the gene AGP1 may not retrieve abstracts discussing this same gene under another name (e.g., YCC5). Still, if the user identifies a relevant document among those returned by the initial Boolean search, PubMed does offer a similarity-based tool (Section 2.3.2) known as *neighboring* (Wilbur and Coffee, 1994) to access documents similar to the relevant one.

While PubMed remains an indispensable resource, clearly a "one-gene-at-a-time" method is not suitable for large scale literature mining. To improve the effectiveness, efficiency, and accuracy of the navigation

[3] *www.ncbi.nlm.nih.gov/entrez/query.fcgi*

through the literature, several methods have been recently suggested, partly automating the literature scanning process. In the following survey, we distinguish between two main directions taken towards meeting this goal. The first, presented in Section 3.1, is based on information extraction and natural language processing. The second, presented in Section 3.2, addresses the literature-mining problem at a coarser granularity and is rooted in information retrieval.

3.1. Information extraction for bioinformatics

Most efforts concerned with biomedical literature mining to date focus on automated *information extraction*, using curated lexica or natural language processing for identifying relevant phrases and facts in text. While we do not go into the details of controlled vocabularies, curated lexica, and ontologies, it is worth mentioning some of the major current sources for gene-related terms: genome and proteome databases such as LocusLink (Pruitt and Maglott, 2001), SwissProt (Boeckmann *et al.*, 2003), and the HUGO gene nomenclature (HUGO, 2003) contain many of the names and synonyms denoting known genes in various organisms. Controlled vocabularies of biomedical terms include the National Library of Medicine's MeSH (Medical Subject Heading) hierarchy (NLM, 2003) and UMLS (Unified Medical Language System) Metathesaurus (Lindberg *et al.*, 1993). The most prominent ontology providing controlled vocabulary for the biological, chemical, and cellular roles of genes and gene products is the gene ontology, GO (The Gene Ontology Consortium, 2000). In the context of bioinformatics, extraction systems typically aim to assist in finding information about a given gene or about relationships between specific genes.

Leek (1997), whose work is the earliest we are aware of in this domain, uses hidden Markov models (HMMs) for extracting sentences discussing gene location on chromosomes. As mentioned earlier (2.1.2), hidden Markov models are often used for representing sentence structure for natural language processing, where states correspond to candidate part-of-speech tags, and probabilistic transitions among states represent possible parses of the sentence, according to the matches of the terms occurring in it to the part-of-speech tags.

In the case of sentences describing gene location on chromosomes, the constituents forming the sentence are gene and chromosome names, words describing location, and terms denoting experimental methods that validate the location of a gene on a chromosome. Names of genes and chromosomes are identified by simple heuristics (e.g., terms in all-capital letters which include numbers are regarded as gene names), and experimental methods as well as localization indicators are provided in a predefined list. The parameters of the HMM itself (namely the probabilities associated with states and transitions) are learned from annotated OMIM (Online Mendelian Inheritance in Man) entries (OMIM, 2000). Both training and test sets consist of several hundreds of sentences. Success is measured in terms of precision and recall (i.e., the number of actual location sentences among those identified as such, and the number of location sentences that were actually retrieved by the system.) Tests on a relatively small set resulted in a success rate of about 0.6 at the break-even point where recall and precision are equal.

Craven *et al.* (Craven and Kumlien, 1999; Ray and Craven, 2001) have extended this line of work, developing systems to distinguish fact-bearing sentences from "uninteresting" sentences. The systems were designed to identify two kind of facts: *protein subcellular localization* and *gene-disorder association*. The earlier work (Craven and Kumlien, 1999) concentrated on training classifiers, with and without the use of grammatical rules, to recognize sentences discussing proteins' location within the cell. Using predefined lexica of locations and proteins and several hundreds of training sentences derived from YPD (Yeast Proteome Database), they train the classifiers and test them over a hand-labeled corpus of about 3,000 PubMed abstracts. The test aims at measuring the ability of the classifiers to correctly distinguish localization sentences from other sentences, rather than to actually extract the subcellular organel for each protein. Without using any grammar-based rules, their highest-precision variation of a naïve Bayes classifier reaches a precision of 77% at a recall level of 30%. A classifier that does use grammatical rules and parsing of sentences reaches a higher precision (92%) but a lower recall (21%). An important result of this experiment is the actual comparison of classifiers to a baseline method, which uses *co-occurrence* alone. The latter method decides that a sentence reports a "subcellular localization" fact if both a protein name and a localization word occur in it. This simple method, which is currently the most popular in the context of literature mining for bioinformatics, reaches a much lower precision than the classifier-based ones at the above recall levels (about 35% precision at recall 30% and 45% precision at recall 21%). The

co-occurrence-based method can reach a higher level of recall (∼70%) without losing much in precision (∼40%). However, at this higher recall level, a naïve Bayes classifier with a noisy-or combination still reaches a somewhat higher level of precision (∼45–50%). The study suggests that classifiers at the sentence level have the potential to improve precision of information extraction, in the biomedical context, over co-occurrence-based methods.

This work was further extended (Ray and Craven, 2001) by using HMMs to represent the sentence structure and to identify sentences discussing gene–disease association. In this case, several hundreds of pre-tagged sentences were used as positive examples for learning the HMMs, while thousands of sentences served as negative examples. No predefined lexica were used, since the relevant terminology is acquired through training from the tagged examples. The correct identification of sentences which refer to explicit genes and proteins is still limited to those containing the names previously used in the training examples.

In the more traditional form of natural language processing through tagging and parsing of sentences, Rindflesch *et al.* (2000) and more recently Friedman *et al.* (2001) propose methods based on parsing and thesauri use to extract facts about genes and proteins from documents. The work by Rindflesch *et al.* is concerned with the effects of drugs on gene activity within the cell, while that by Friedman *et al.* targets interactions among genes and proteins as part of regulatory pathways.

A simpler approach that relies on co-occurrence of genes/proteins within sentences, rather than on machine learning methods or advanced NLP, was used by Blaschke *et al.* (1999). Its goal was to extract information about protein interactions among a predefined set of related proteins from scientific text pertaining to them. Using a list of protein names and a list of interaction words, they look for sentences that have occurrences of two protein names separated by an interaction word, to identify relationships among the proteins. An extension to this work is described by Blaschke and Valencia (2002), where they use a module for protein name detection (an issue we touch on briefly later) and exclude negations. The latter means that interaction facts are extracted only from sentences that affirmatively report the interaction.

The exclusion of negation is an interesting point and merits some discussion. The concern about negation sentences (e.g., "We have found *no evidence* that protein A is involved in the regulation of gene B") is often expressed in the context of mining the biomedical literature. The assumption underlying this concern is that we want to avoid, for instance, relating protein A and protein B in a regulatory pathway if according to the literature the two are not related. This is indeed a valid point if we aim to automate the construction of pathways through the literature. However,[4] under different scenarios, for instance, when investigating a set of proteins and genes in which protein A is produced just before gene B is expressed, an edge between A and B marked with a "negative regulation" label and linked to the relevant article stating the negative result is extremely valuable. Hence, the reconsideration of negation, its role, and its treatment is pertinent.

While all the methods discussed so far have typically been applied to small sample sets of documents and entities, a major step towards large-scale analysis was recently taken by Jenssen *et al.* (2001). Using a *predefined* list of gene names and symbols, they executed a Boolean search over PubMed, finding *all* abstracts in PubMed mentioning these genes. They then built a graph with the genes as nodes and edges connecting genes that are mentioned in the same abstract. Weights on the edges represent the number of co-occurrences. The result is a very large network of genes related through the literature and abstracts justifying each edge. Jenssen's work is the most extensive effort to date towards using the literature on a genome-wide scale, providing an unprecedented tool for researchers.

Numerous other co-occurrence-based systems have recently been reported in the literature, all concerned with information extraction of facts about biological entities from biomedical text. The essential commonality among them is that they all look for co-occurrences of names or identifiers of entities, typically along with activation/dependency terms. The differences between the systems are typically in the extent to which they use syntactical analysis and other NLP methods, as well as the vocabularies and thesauri that they utilize (Tanabe *et al.*, 1999; Thomas *et al.*, 2000; Stapley and Benoit, 2000; Humphreys *et al.*, 2000; Park *et al.*, 2001; Yakushiji *et al.*, 2001; Hahn *et al.*, 2002; Pustejovsky *et al.*, 2002).

However, all of the above methods suffer several limitations. A relatively minor one is that they all require the specification of a very accurate query, using the explicit names of genes (or other entities of interest) to provide high-quality results. A more prominent issue is that they all rely on strong assumptions

[4]Thanks to Mark Novotny for making this point explicitly.

about the use of natural language, such as terms typically used to indicate relationships, the typical sentence structure, gene/protein names and their format, and the way these names are used within sentences. Such assumptions are not readily met throughout the abundant biological literature, (see Pearson [2001] for an extensive discussion), thus limiting the scope within which these methods are effective. The methods also strongly rely on having a *complete list of gene names and synonyms*.

Moreover, since these methods depend on the co-occurrence of terms, within a sentence, a phrase, or an abstract (see Ding *et al.* [2002] for a discussion on granularity choice), they can only reveal relationships that are *already* reported in the literature and do not attempt to detect new relations. We qualify this with the observation that one could follow Swanson's methodology (Swanson, 1986, 1988, 1990), and use the "transitive" relations—i.e., the indirect-links among entities—as clues for yet-unknown relationships. For instance, if there is a report relating protein A to B, and another report relating B to C, it may suggest a possible (yet-unreported) relation between proteins A and C. It is also important to note that as large-scale experiments using microarrays and other high-throughput techniques are becoming more popular, the co-occurrence of gene and protein names in the literature may become more of an indicator of their inclusion in large-scale experiment rather than of an actual functional relationship between them. When using the literature to interpret the results of large-scale experiments, it is crucial that the literature-mining engine could actually provide an independent insight into the functional and biological relationships—beyond the mere fact that they participated together in a large scale experiment. Methods that strongly rely on co-occurrence alone are insufficient to address this need.

As mentioned before, many of the above systems rely on finding the occurrences of protein/gene names within the text, typically based on a lexicon that was gathered from one of the public databases. Automated protein/gene name detection in text is an active research field in and of itself. The reader is referred to the early method developed by Fukuda *et al.* (1998) and to the recent work by Hanisch *et al.* (2003) and references therein.

The above discussion demonstrates that a lot of effort is put towards using methods from information extraction and natural language processing. These methods strongly rely on predefined information, much of which is hard to obtain or to take for granted (e.g., agreed-upon nomenclature in genomics and proteomics). A way to relax these requirements is much needed. An alternative, or a *complement*, to the fine-granularity approach of using explicit gene names/synonyms while searching for relationship sentences or co-occurrences, is the search for relevant abstracts. Information retrieval, which inherently works at a higher level and at a coarser granularity, namely that of documents and abstracts, has therefore much to offer. In the following section we discuss its application.

3.2. Information retrieval for bioinformatics

The most common and simple form of information retrieval is already broadly and regularly used by researchers searching for articles of interest. As discussed in the beginning of this section, PubMed supports both Boolean queries, based on Boolean combination of keywords, and a limited form of similarity queries. While PubMed is an effective tool for retrieving well-targeted documents of interest, it is not intended and cannot be used "as is" for finding or explaining large-scale relationships among genes or other biological entities. However, methods to support large-scale biomedical analysis based on information retrieval have been introduced and developed.

The earliest extensive work we are aware of in this direction is by Shatkay *et al.* (2000, 2002). Its goal is to find functional relationships among genes, without strongly relying on gene nomenclature or on sentence structure. The approach is based on the hypothesis that many *individual* genes and their function are already discussed in the literature. We use a large[5] collection of PubMed abstracts, covering the literature relevant to the domain of discourse (e.g., *all*) the abstracts in PubMed discussing yeast genes, or all the available abstracts discussing genes participating in a specific microarray experiment). To find relationships among a large set of genes, we map each gene to a single abstract within the collection, discussing the gene's biological function. This abstract is treated as the gene's *representative* and is called the *kernel abstract* for that gene.

[5]On the order of several *tens of thousands* of abstracts.

We then use a probabilistic algorithm (Shatkay and Wilbur, 2000) that given an example document, finds a set of documents most relevant to it (*a theme*) and produces a *set of terms* summarizing the contents of this document set. Applying this theme-finding algorithm to each kernel produces for each gene a body of related literature (20–50 abstracts bearing a common *theme*) based on the kernel abstract representing it, along with a list of terms that characterize the relevant literature. Once a set of abstracts is retrieved for each gene, we use an automated method to compare the abstract sets and derive functional relationships among genes. This method was tested on a collection of hundreds of yeast genes over a database of about 40,000 PubMed abstracts; a thorough study of the biological roles of yeast genes by Spellman *et al.* (1998) was used for validation. This work is described in more detail in Section 4.1.

Several other groups have recently applied information retrieval methods, mostly variations on clustering and classification, in the context of bioinformatics. Renner and Aszódi (2000) suggested a method for clustering protein annotations, the basic idea being that by clustering the annotations of proteins into groups one can gain insight into the common function that the proteins may have. The method is based on first grouping terms that occur in protein annotations—or in any document in the general case—into sets, according to their tendency to co-occur. A similarity measure among documents is then devised based on the proportion of terms in them that are in the same term groups. Finally, the annotations are clustered using hierarchical clustering based on the suggested similarity measure.

Iliopoulos *et al.* (2001) apply k-means clustering to a relatively small set of PubMed abstracts (less than 2,000 documents) to obtain meaningful subsets, each discussing some common subject. The subjects are then represented through terms extracted by statistical analysis of term frequencies within the clusters. Marcotte *et al.* (2001) apply a Bayes classifier that relies on discriminating terms to identify abstracts that discuss protein–protein interactions.

While the above methods basically categorize a (relatively small) collection of abstracts into useful topical sets, the methods we survey next apply other information-retrieval techniques to directly enhance, through the use of text, specific applications that typically rely on biological data.

Using an idea similar to the one presented earlier (Shatkay *et al.*, 2000) of viewing an abstract as a "surrogate" or a representative for a gene, Stapley *et al.* (2002) represent proteins using the abstracts that mention them. They then train a support vector machine (SVM) classifier to distinguish among abstracts discussing proteins, based on the different subcellular locations of the proteins mentioned in the text. They propose this classifier as an aid in addressing the protein-sorting problem, which is the task of determining the organel within the cell to which the protein belongs. (Note that the same task was tackled through information extraction by Craven *et al.* as presented before.)

Stephens *et al.* (2001) deduce relationships among genes based on co-occurrence of their names (where the names are given in a thesaurus), which is a subject already surveyed in Section 3.1. However, unlike the methods mentioned earlier, this work uses information retrieval techniques to identify co-occurrence. Documents are represented as weight vectors, where the genes mentioned in the documents are the terms. By considering the transposed matrix, each gene is viewed as a vector whose attributes reflect the documents that mention it. The association between genes is measured by the dot product between the two vectors representing them (which corresponds to the unnormalized cosine coefficient). This measure effectively quantifies the co-occurrence of the genes within documents.

Another application of an information retrieval technique is in the realm of protein homology, as shown by Chang *et al.* (2001). In this work, text that accompanies protein sequences is used to improve homology search. While PSI-BLAST is applied to the protein sequences to detect homology among proteins, it is augmented with the cosine similarity measure which is applied to the accompanying text.

Most recently, Donaldson *et al.* (2003) introduced their PreBind/Textomy system, in which they combine information retrieval with information extraction to assist in recovering protein–protein interactions from the literature. On the information retrieval step, an SVM classifier is trained to distinguish between PubMed abstracts that discuss protein–protein interaction and abstracts that do not. The classifier is then used to identify and retrieve the abstracts that are relevant to protein–protein interaction. Once they are retrieved, information extraction is applied to identify interaction facts within the text. The SVM is used here again to find sentences containing the interaction information. From each such sentence, protein names are extracted (based on a list of protein names and synonyms), as candidates for protein–protein interaction. Simple co-occurrence of protein names within the complete abstract is also an indicator for possible interaction between the proteins. The system, which serves as a curation aid for the BIND database (Bader

et al., 2003), also provides a user interface in which the potential interactions are identified (by automated application of pattern-matching rules) and highlighted. Thus, the putative interactions found are not meant to be automatically placed in an interaction network. Rather, the BIND curators examine and validate them by reading the related text.

The retrieval system was trained and tested, using 10-fold cross validation, on a set of about 1,100 expert-judged abstracts of which ∼700 discuss interactions and ∼400 do not. The success rate reported is 92% in both precision and recall for identifying abstracts discussing interaction. The extraction of actual protein–protein interactions was tested by comparison to a list of about 1,380 human-curated protein–protein interactions in yeast (restricted to interactions reported in the literature). About 60% of these interactions were successfully recovered from the abstracts that were classified as discussing interactions. Of these, only about 1/3 were recovered from a sentence identified as an interaction sentence by the SVM and containing both protein names, while 2/3 were recovered based on co-occurrence of the protein names anywhere in the abstract. The higher accuracy of the early classification phase suggests that a major advantage of the system lies in the retrieval step which identifies documents relevant to protein–protein interaction.

4. LITERATURE MINING SYSTEMS—EXAMPLES

This section describes two systems that utilize the biomedical literature. The first, GenTheme, is based on information retrieval (Shatkay *et al.*, 2000, 2002) while the second, LitMiner, is an information extraction system which was the winning system in the KDD-cup 2002 competition (Regev *et al.*, 2002; Yeh *et al.*, 2002).

4.1. Information retrieval for functional relationships among genes

The goal of the work described here is to find functional relationships among genes on a large scale, that is, over hundreds or thousands of genes, primarily in the context of large-scale expression experiments, but not limited to it. It relies on the fact that many individual genes and their function are already discussed in the literature. The main assumption behind this method is that common function-related relevant literature is a strong indicator of common function among genes. Thus, the method consists of two phases:

- Finding the relevant body of literature—which we call a *theme*—for each gene.
- Detecting commonalities between the bodies of literature associated separately with each gene. These commonalities are used to relate genes to one another.

4.1.1. Themes for genes. The search for themes is conducted within a large[6] collection of PubMed abstracts covering the literature relevant to the domain of discourse (e.g., *all* the abstracts in PubMed discussing yeast genes). Each gene is initially mapped to a single abstract within the collection discussing the gene's biological function. This abstract is treated as the gene's *representative* and is called the *kernel abstract* for that gene.

A theme-finding algorithm, which searches for abstracts relevant to a *kernel abstract* (see Shatkay and Wilbur [2000] for a complete discussion), is used to establish the body of literature for each of the genes. The idea underlying this algorithm is that a set of documents sharing a coherent *theme* can be characterized by a set of probability distributions. For example, documents discussing genes responsible for *nutrition* during the cell cycle are likely to contain terms such as *fructose* or *glucose* and unlikely to contain the term *lipid*, as illustrated in Fig. 6. We use mixtures of Bernoulli distributions, which are in turn represented by sets of distribution parameters.

The algorithm uses an iterative optimization method based on *expectation maximization* (EM) to find the theme parameters. The result, in addition to the model parameters themselves, is a ranking of the

[6]On the order of several *tens of thousands* of abstracts.

FIG. 6. Typical term distribution for the *Nutrition* theme.

documents by relevance to the theme of the original kernel document, as well as characteristic terms. As an output it produces for each gene

- a body of related literature—i.e., a list of the top 50 documents discussing the same theme as the kernel document, ordered by their degree of relevance to the theme, and
- a list of terms (keywords) characterizing this relevant literature, ordered by their degree of relevance to the theme.

The process is depicted at the top of Fig. 7.

It is important to note that, in contrast to other literature-based methods discussed in Section 3, the retrieved abstracts are considered relevant *not* because they contain the *same gene name* as the one associated with the kernel abstract, but rather because they discuss the same *topics* (in this context, typically related to a specific function) as those discussed in the kernel abstract.

While this output in and of itself provides valuable support for gene analysis, it is further extended, in a follow-up phase, to assist in finding biological relations among the genes. Once a set of abstracts is retrieved for each gene, we use an automated method to compare the abstract sets and derive functional relationships among genes, as described below.

4.1.2. Finding functional relations among genes. The underlying assumption here is that common relevant literature indicates common function among genes; genes with similar associated lists of top-ranking documents share some common biological function described in the common literature. The task

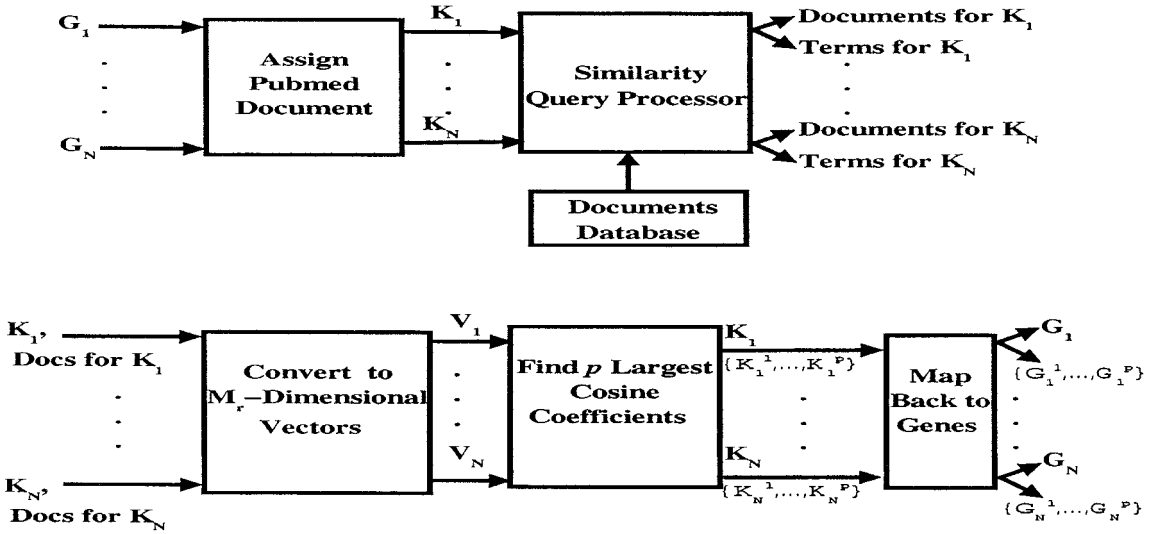

FIG. 7. Finding documents and terms related to genes (**top**), and sets of related genes (**bottom**).

is thus reduced to comparing the *sets* of documents retrieved in the previous phase of the algorithm and associating with each gene all other genes that have a similar document set.

To do this, we use the *PubMed identifiers* (*PMID*s) associated with the abstracts, rather than the actual abstracts' contents. For each kernel, we construct a characterizing *vector*, consisting of the *PMID*s of the abstracts found relevant to it by the first phase of the algorithm. Note that this vector is not a term vector as was presented in Section 2.3.2 since its entries represent *associated abstract identifiers* rather than *terms*. Still, by using the cosine measure with respect to these vectors, we can rank for each kernel K_i all the other kernels by their proximity to K_i in the *PMID*-vector space. Since each kernel corresponds to a gene, we can map the interrelated kernels back to their respective genes and obtain a set of genes that are closely related. The method is illustrated at the bottom of Fig. 7.

The experiments and the results reported next demonstrate the value of the methods for retrieving relevant abstracts and terms and for obtaining meaningful relationships among genes.

4.1.3. Experiments and results. To test the method, the algorithms were applied to yeast genes, since several available data sources in this area allowed the validation of the results (the SGD, Saccharomyces Genome Database [Dolinski *et al.*, 2000], the YPD, Yeast Proteome Database [Hodges *et al.*, 1999], as well as the functional analysis given by Spellman *et al.* [1998]). Specifically, the quality of the results was verified through comparison to the functional groups of genes according to Spellman *et al.* (1998). The portion of Spellman's table relevant to the results discussed here is shown in Table 1. The table categorizes the yeast genes according to their function (rows)[7] and the phase in the cell cycle in which they are expressed (columns). The algorithms were applied to 408 cell-cycle-regulated yeast genes for which curated PubMed references were available in the SGD. For each of these genes, the oldest reference cited in the SGD was used as the *kernel abstract* corresponding to the gene. Since some of the closely related genes share the same reference, we obtained 344 distinct kernel abstracts. The database used is a subset of PubMed, consisting of 33,700 abstracts discussing yeast genes.

In the first phase, the theme-finding program was applied to the 344 kernels, searching over the database of 33,700 abstracts. The results for each kernel consist of

1. a set of related abstracts,
2. a set of summarizing key terms.

In the next phase, we obtain from the set of related abstracts for each kernel, through the vector representation and the cosine coefficient calculation, a *set of related kernels*. The latter are mapped back to their respective genes, forming a *set of related genes*. The sets of genes that were grouped as similar according to our method were then compared with those grouped by function in Spellman's table.

As a *qualitative assessment* we examine the set of summarizing keywords and the lists of related genes obtained by our method and compare those with the function assigned to genes by Spellman *et al.* An example of a typical successful search is shown in Table 2. The left column lists the PubMed identifier for the kernel abstract along with the gene it stands for and its function according to Spellman *et al.* The second column lists the top 10 keywords associated with the retrieved set of abstracts, as determined by the algorithm. The third column lists the top genes associated by the algorithm with the kernel, based on the cosine coefficient. The fourth column lists the function of each gene according to Spellman *et al.*, as a validity check for our results. Since the experiment included more genes than listed in Spellman's table, some of the genes in the third column are not assigned function by Spellman. For these genes (marked by "*" in the table), the function is assigned according to YPD.

The table shows that almost all the genes found for the kernel have a strong functional relationship to the gene represented by it, and the keywords provide a strong indication of this biological function. We note that PGM1 is involved in carbohydrates metabolism which is still functionally related to fatty acids metabolism. The results for about 100 out of the 220 kernels for which we had the Spellman-assigned function closely resemble the ones demonstrated in Table 2 in strong agreement with Spellman's cluster assignment and

[7]The gene function assigned by Spellman *et al.* is based on human judgment and a lot of expertise, rather than on an automated process.

TABLE 1. YEAST GENES: EXPRESSION DURING CELL-CYCLE AND FUNCTION. (ADAPTED FROM SPELLMAN *et al.* [1998])

Biological function	*G1*	*S*	*G2*	*M*	*M/G1*
Replication initiation	CDC45		ORC1	CDC47 CDC54 MCM2 MCM6	CDC6 CDC46 MCM3
Fatty acids/lipids/sterols/ membranes	EPT1 LPP1 PSD1 SUR1 SUR2 SUR4		AUR1 ERG3 LCB3	ERG2 ERG5 PMA1 PMA2 PMP1	ELO1 FAA1 FA A3 FAA4 FAS1
Nutrition	BAT2 PHO8		AGP1 BAT1 GAP1	DIP5 FET3 FTR1 MEP3 PFK1 PHO3 PHO5 PHO11 PHO12 PHO84 RGT2 SUC2 SUT1 VAP1 VCX1 ZRT1	AUA1 GLK1 HXT1 HXT2 HXT4 HXT7

Table 2. Example of a Result Obtained Using Our Algorithm, Compared with Function According to Spellman or YPD (YPD Function Denoted by *)

Kernel (PMID, gene, function)	*Keywords*	*Assoc. genes*	*Function*
8702485	fatty acid,	OLE1	(fatty acid, sterol. met.)*
ELO1	fatty,	FAA4	fatty acid/lipids/sterols/membranes
fatty acid/lipids/sterols/membranes	lipids,	FAA3	fatty acid/lipids/sterols/membranes
	acid,	SUR2	fatty acid/lipids/sterols/membranes
	grown,	FAA1	fatty acid/lipids/sterols/membranes
	medium,	ERG2	fatty acid/lipids/sterols/membranes
	carbon,	PSD1	fatty acid/lipids/sterols/membranes
	synthase,	CYB5	(fatty acid, sterol. met.)*
	strains,	PGM1	(carbohydrates met.)*
	deficient		

in the accurate description as given by the keywords learned by the similarity query algorithm. For other kernels, the groups of related genes contain many genes not assigned function by Spellman, which makes the results harder to validate. Another culprit in assessing the strength of the method using this data, as was noted in earlier publications (Shatkay *et al.*, 2000, 2002), is that the kernel abstract we use—which is the oldest curated reference for the gene in the SGD—often does not discuss the biology of the gene but rather the experimental technique used to discover it. In this case, many other genes whose kernel abstract is also related to the technique rather than to the gene's function are found to be related to each other, but this relationship is not function based. Typically, the keywords indicate and expose the problem. Still, in the general case, it is desirable to have an automated method for kernel picking, based on a classifier that would distinguish abstracts that discuss function (and make "good" kernels) from those that discuss experimental methods ("bad" kernels).

To *quantitatively* measure the validity of the keyword list assigned to each kernel, we compare each keyword to its associated function using a mini-thesaurus obtained from a panel of four independent yeast experts. Each function descriptor listed in Spellman's table (such as *Secretion* or *Chromatin*) is associated with the terms judged most closely related to it according to the experts. To construct this thesaurus, each expert received a list of the 22 function descriptors listed by Spellman *et al.*, and a separate list of 330 alphabetically sorted summary terms resulting from our program. The experts assigned to each term in the latter list the function descriptors that they judged to be most related to it; nonspecific terms were left unassigned. An example of two entries in the resulting thesaurus is shown in Table 3.

Running on a collection of 100 genes, each with a biologically-informative kernel abstract, the true function for each gene according to Spellman was compared with the function matched by the panel for each of the key terms assigned by our method, counting how many of the key terms indeed correctly describe the gene's true function. A keyword occurring in the term list generated by our method for a specific gene is considered *correct* if it appears in the thesaurus entry labeled by the same function as the one assigned to the gene by Spellman. If its thesaurus entry is labeled by a different function, it is considered *wrong*. If it was assigned no function by our panel of experts it is considered *nondescriptive*. An average was then taken over all 100 genes of the number of correct (versus incorrect) keywords among the five top-ranking keywords associated with each of the 100 kernels. On average 3.27 out of the five

Table 3. Example of Thesaurus Entries Associating Gene Function with Related Terms

Function descriptor	*Associated terms*
Chromatin	chromatids, chromatin, chromosome, sister chromatids, telomere, telomeric
Secretion	acid phosphatase, coatomer, endoplasmic endoplasmic reticulum, er, golgi apparatus golgi complex, golgi transport, golgi, v snare

top-ranking keywords, were associated with the correct function, while only 1.12 out of the five were associated with the wrong function, and 0.61 out of the five were nondescriptive.

The above demonstrates several clear advantages of the information-retrieval-based approach:

1. It is an effective way for detecting putative relationships among genes. These can then be verified through well-targeted experiments.
2. It provides the relevant literature for analyzing experimental results.
3. It generates summarizing terms explaining the discovered relationships. This summary can help explain and evaluate the relationships found by direct clustering of the expression levels.
4. It is independent of natural language usage and nomenclature issues, as it does not search for explicit gene names or statements about their relationships.

We also note that the method neither uses any pre-clustering of the genes among which it is looking to find relationships, nor does it make assumptions about the use of natural language in the documents or rely on the availability of curated/agreed-upon nomenclature. It is expected that with a good source of kernels, utilizing *multiple kernels* for each gene, rather than a single kernel, we can obtain a better initialization for the EM algorithm and further improve the results. Another promising direction is that of extending the vector representation of abstracts to include gene expression values, simultaneously searching for related abstracts and similarly expressed genes.

Throughout the rest of this section, we provide another example, describing the use of the information-extraction system LitMiner for identifying text relevant to gene expression.

4.2. LitMiner: Information extraction of gene expression data

The 2002 KDD-cup competition (KDD, 2002; Yeh *et al.*, 2002) assigned a task dealing with the information curation process within the FlyBase consortium (2002). One of the challenges facing FlyBase curators is to find, among the published Drosophila-related scientific literature, specific evidence for experimental results regarding products of the wild-type Drosophila genes. The curators search for sentences and phrases indicating a report of a new experiment resulting in a gene expression product. When a paper containing such phrases is found, they store its identifier and the evidence within it in a database, along with an indication whether the expression evidence pertains to a transcript or to a protein product.

The KDD task was to create an automated system that, given a set of full-text papers as well as a list of gene names and synonyms (per paper), does the following:

- Identifies which publications indeed contain experimental reports as described above of gene expression (for genes within a gene list associated with the paper) and orders the papers according to their relevance to expression experiments in Drosophila genes.
- Specifies which of the genes mentioned in the paper had their expression actually discussed in it.
- Distinguishes for each expressed gene whether the reported expression product is a transcript or a protein.

Note that while the human curators rely in their judgment on figures that are part of the original papers, the version provided for the KDD-cup consisted of text alone.

The first task can be viewed as a text-categorization problem: Given a scientific paper, classify it as either *curatable*—if it includes an experimental expression evidence—or as *non-curatable* otherwise. Even the third task that requires a decision regarding the type of product (transcript or protein) of each gene can be presented as a categorization task (Ghanem *et al.*, 2003). On the other hand, as pointed by Yeh *et al.* (2002), information extraction is less intuitive in this context, since it usually deals with the local extraction of specific template instances and not with the higher-level task of document relevance judgment. Still, previous work has shown that information extraction can be used to improve the retrieval of relevant documents (Bear *et al.*, 1997).

Moreover, closer analysis of the tasks and actual experiments with the provided text demonstrated that the information extraction approach was indeed suitable for this task for the following reasons:

1. Most papers were from the same limited domain of Drosophila-related molecular biology and genetics, using a common narrow vocabulary. Distinguishing relevant from irrelevant documents based on termi-

nology, which is what most categorization approaches do, becomes very hard when the documents are so similar in their term distribution.

2. Many curatable papers contain both relevant results (wild-type expression) and irrelevant ones (mutated-gene expression). Moreover, the paper may include results relevant for some of the genes mentioned in it but not for others. Classical categorization cannot, therefore, realize the task of identifying which of the genes in the associated gene list are reported to be expressed in the paper. The latter requires relevant phrases (i.e., patterns) for the specific genes to be extracted and examined.
3. The third part of the task requires a decision, for each gene, whether it was expressed in terms of a transcript or a protein. In many cases, there is only a fine distinction in the language used to discuss the two product forms. For instance, lower-case letters are used for naming proteins while upper-case for naming genes. Often a protein product is only implied through the discussion of the transcript (or vice versa). Hence, an accurate analysis of the phrases is needed, and the preferable choice in this case is the use of an information extraction, rule-based system.
4. While not explicitly required for the task, an information extraction system which identifies the relevant phrases within the text as a basis for making the relevance judgment provides the user with evidence justifying the relevance decision. This allows the user, or the system developer, to analyze the reasons behind each relevance judgment and to further refine/improve the system as needed.

For the KDD task, we therefore used the LitMiner extraction system as described next.

4.2.1. A hybrid tagging approach. The LitMiner system processes documents by using a multi-strategy tagging approach. It combines statistical tagging (categorization), semantic tagging (information extraction), and structural tagging (visual layout analysis). Figure 8 presents the architecture of the LitMiner system. The typical input to the system (bottom) are web documents, but other text sources, such as *word* files or PDF files can be used as well. The output of the tagging phase is a rich XML (or DAML) document, which can be fed into either enterprise systems, such as work flow systems, file systems, and corporate databases, or into dedicated business intelligence suites (which are basically sophisticated search engines).

The architecture for a hybrid tagging system is shown in Fig. 9. The training module for the statistical tagging produces classifiers for each of the predicted categories, while the training module for the semantic tagger produces information extraction rules based on an annotated training set of documents. The rules identify entities and the relationships between them. The semantic tagger may be further customized by writing additional information extraction rules, using the DIAL language described below. We focus here on the semantic tagger and elaborate on the information extraction mechanism.

The LitMiner system searches for relationships among entities (e.g., *genes, proteins, transcripts*). The input to the system is a set of biomedical articles, typically drawn from a broad subdomain. These articles

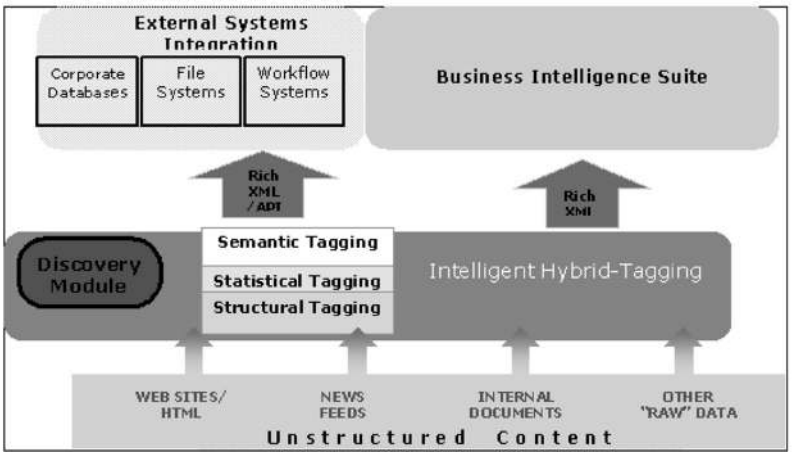

FIG. 8. Architecture of the LitMiner system.

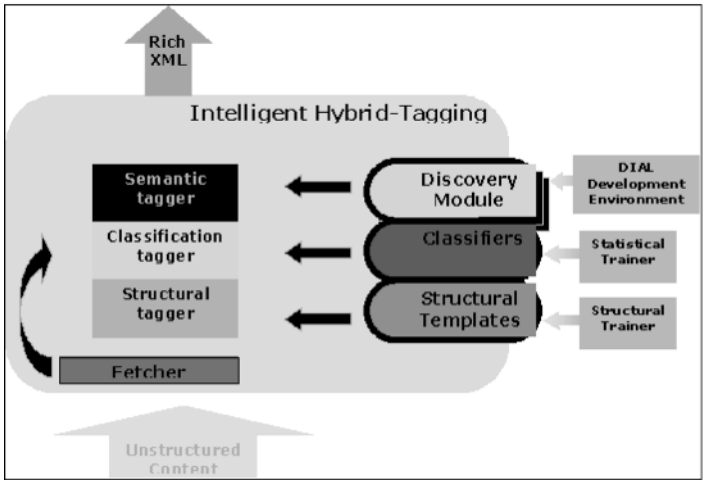

FIG. 9. Architecture of the hybrid tagging system.

are analyzed by the information extraction module, and all entities and relationships are extracted from them. We describe here the modules which identify the entities and relationships of interest within the articles.

4.2.2. Defining information extraction modules. For the purposes of the KDD-cup, rather than search for relevant information throughout the whole document, we concentrate only on the elements on which a human reader, scanning quickly through the paper, would focus. Following discussions with domain experts, we identified the following relevant (and easily identifiable) parts within articles:

- Figure captions. These seem to explicitly state the experimental results found, using a relatively limited and concise set of patterns and vocabulary.
- Titles. Relevant papers seem to have specific relevant keywords or patterns (such as *Expression of a Drosophila GATA Transcription Factor*) within their titles, while the titles of irrelevant papers may contain keywords such as *ectopic* or *interference* that were identified as "negative" for the particular task of identifying expression of wild-type naturally occurring genes.
- Abstracts. These summarize the major finding of the paper.
- Footnotes. These often contain indications that the author submitted a novel sequence to GenBank (e.g., "*The nucleotide sequence(s) reported in this paper has been submitted to the GenBank(TM)/EMBL Data Bank with accession number(s) D50542[GenBank].*").

To extract information from the text, key concepts, namely *entities* or *relationships* between entities, are defined in advance and the text is searched for concrete evidence for the existence of these concepts. In the case of the KDD-cup, the *entities* were the gene names and their products (proteins and transcripts) while the relationships were the textual indicators for transcription or translation. When a phrase containing these entities and relations was found, the relevant attributes were extracted from that phrase and stored in a structured format. Thus, structured information was created from the unstructured text. To facilitate this transformation, specific rules that support the identification of the relevant phrases were used as explained below.

4.2.3. Structure-driven rule-based strategy. The strategy of rule-based extraction is based on identification of natural language elements (noun phrase and verb phrase) augmented by linguistic and semantic constraints. Under this strategy, the extraction of the predefined semantic relationships is performed by means of deep syntactic and semantic analysis of the sentences.

The implementation of the structure-driven processing is based on a general multilayer natural language processing (NLP) system. As detailed below, the multilayer architecture enables easy adaptation for new

entities and relationships or even a new domain. While the lexical and semantic sources for domain-specific entities or relationships need to be replaced for a completely new domain, the overall structure typically remains unchanged. Here is a brief description of the various layers:

Layer 0 Part of speech (PoS) tagger: Assigns PoS tags (e.g., *noun, proper noun, verb, adjective*, etc.) to each word. For instance, in the fragment "*. . . expression of dGATAc transcripts. . .* ", the terms *expression* and *transcripts* are nouns while *dGATAc* is a proper noun.

Layer 1 Noun phrase and verb phrase grouper: Groups the head noun with its left modifiers (for example: "*the developing midgut*") and for verbs, groups a main verb with its auxiliaries (e.g., "*does not antagonize*").

Layer 2 Verb and noun pattern extractor: Extracts larger verb and noun phrases, based on semantic requirements, for instance, "*Dac does not antagonize Dll expression*."

This level is semantically oriented: it keeps track of the semantic features of a pattern as expressed by various elements such as adverbs, tense and voice of the verb group, and certain syntactic structures.

Layer 3 Named entity recognizer: Recognizes the entities relevant to the domain. In the context of the KDD-cup, these entities include genes, proteins, transcripts, etc. Other domains will naturally have different entities; for example, typical entities in the financial news domain include companies, people, products, and so forth.

Layer 4 Template (*Relationship*) extractor: Rule-based extraction of patterns at a full sentence or phrase level, using the components identified by previous layers. For example, in the KDD competition, it was required to separate evidence for gene expression in the wild-type genes, such as "*Figure 2. Northern blot analysis of fruitless mRNA*," from evidence for expression induced by artificial intervention or mutation of the genes, as in the caption "*Figure 3. Ectopic expression of dNSF2 in mesoderm is sufficient to rescue the lethality of dNSF2 mutations*." To extract such complete instances, we combine the NLP tools and semantic constraints with lexical resources. The latter are used, for instance, to identify the term "*northern blot*" as a technique for recognizing transcripts, and *fruitless* as a gene name.

4.2.4. Implementation in the DIAL language. The framework used for implementing the information extraction system is the rule-based general IE language DIAL (declarative information analysis language) (Feldman *et al.*, 2001). Rules in DIAL are sequences of pattern-matching elements, augmented by a set of constraints that matched patterns must satisfy, as well as by a set of assignments to rule's parameters and actions regarding external data structures. The pattern-matching elements are either explicit strings found in the text (such as the word *expression*), a word class (a specific set of lexicon terms), or another rule.

While the complete syntax of DIAL is beyond the scope of this paper, the main feature to note here is that DIAL enables the user to implement separately the various operations required for performing IE. These operations include tokenization, sectioning (recognizing paragraph and sentence boundaries), morphological/lexical processing, parsing, and assigning domain semantics. DIAL has built-in modules that perform the general tasks of tokenization and part-of-speech tagging. Customized rules are added as needed for handling domain-specific entities and sentence structure. An IE module, called a *rulebook*, incorporates the infrastructure libraries and specific customized rules for a specific domain or task.

Figure 10 demonstrates a sample rule used in the KDD-cup. The rule identifies *induced expression*, that is, an expression (or lack of expression) of a gene, reported under certain conditions that are influenced by another gene. Such expression should not be extracted as a natural, wild-type expression, and the rule is used to distinguish expression phrases that justify curation of a document from phrases that do not. We use a lexicon (*wordclass*) of expression nouns (*wcExpressionNoun*) and a lexicon of verbs indicating such a relation between two genes (*wcInducedVerbs*). The predicate *ExtractedGene* is implemented in the named-entity-recognizer layer, where a gene name is matched and mapped to its canonical form. Induced expression is the main rule in this example; when a gene is found, followed by a verb group whose main verb is in the *wcInducedVerbs* wordclass, followed by an expression of another gene (*GeneExpressionNG*), the phrase is an instance of induced expression. Therefore, it should *not* be treated as a positive evidence for either gene despite its discussion of both the genes and their expression.

DIAL Rule example:
Induced expression: The gene expression is induced or induces another activity.
That is - the gene expression is mentioned in the text but is NOT observed on its own, as in:
"Fig. 4. Dac does not antagonize hth expression in the antenna".

```
//lexicon for relevant nouns similar to "expression":
wordclass wcExpressionNoun = expression transcription localization detection;

//lexicon for verbs indicating induction/interaction between genes such as "antagonize":
wordclass wcInducedVerbs = reduce inhibit activate induce repress alter antagonize;


//extract Noun Phrase (NG-Noun Group) incorporating a gene
 GeneExpressionNG() :-
 ExtractedGene(Gene, Product)              //"hth" (The gene)
 NounGroup(Article, Head, Stem)            //"hth expression"
 verify(InWC(Head,@wcExpressionNoun));     //verify that the Head is relevant

//Rule for the Induced Expression itself
 InducedExpression() :-
 VerbGroup(Stem,Tense,Aspect,Voice,Polarity)    //verb group-"does not antagonize"
 GeneExpressionNG() :-                          //"hth expression"
 verify(InWC(Stem,@wcInducedVerbs);             //verify that the Stem is indeed a
                                                //relevant verb
```

FIG. 10. Dial rule example.

4.2.5. System performance. We developed our system based on a training set of 862 full text articles tagged by the FlyBase curators, provided to the KDD-cup participants. The system was tested on the KDD separate set of 213 articles (See Yeh *et al.* [2002] for more information).

Our system achieved the best results in all three tasks, ranking the highest among 32 participating systems. The results were evaluated based on the simple F-score (see Section 2.5). Our F-score result for the second task—curate/do-not-curate decision—was 78% (compared with a median of 58% for all the systems submitted). Our F-score result for the third task—expressed/not-expressed decision regarding each gene—was 67% (compared with a median of 35% for all the system submitted). While there are obvious limitations to the evaluation criteria, such as the small size of the test set, these results still clearly demonstrate the superiority of the rule-based IE approach for this task over the variety of other methods used by the different entries in the competition.

5. CONCLUSIONS

The abundance of biomedical literature motivates an intensive pursuit for effective text-mining tools. Such tools are expected to help uncover the information present in the large and unstructured body of text, while addressing three main problems:

- the sheer magnitude of the available text collections;
- the ambiguity and nonuniformity of the nomenclature used in the context of genomics and proteomics;
- the linguistic-complexity of the scientific documents, stemming from the diversity of the authors in terms of expertise, style, and native languages.

We have surveyed the prominent methods used for text mining and demonstrated their application in the context of biomedical literature mining.

In general, information retrieval provides the means for a coarse-grain search for relevant documents. While it is not intended to extract a tidy fact statement, it can produce a relatively small set of choice documents, thus restricting the search space within which the facts of interest can be found. This focused set of documents can also provide the relevant literature needed for analyzing and explaining experimental

results (on which other automated mining systems may operate). Moreover, we have demonstrated that a nontraditional use of information retrieval can actually provide an effective way for detecting specific putative relationships among genes, while generating "snippets" of summarizing terms explaining the discovered relationships. Since information retrieval does not look for explicitly stated facts within the literature, it has the potential to *foreshadow yet undiscovered facts*. The most important advantage of the information-retrieval approach is its relative independence of specific natural language usage and nomenclature issues, as it does not search for explicit gene names or statements about their relationships. The latter is a major feature given the complex and incomplete nomenclature of the biomedical domain.

On the other hand, we have demonstrated that for fine-grain discovery, information extraction has significant strengths. While it does rely on vocabularies, lexica, and assumptions about language structure and usage, it can provide a flexible and extensible framework for detecting facts and relationships within specific documents or document sections. We have also shown (using the KDD-cup example) that the nontraditional use of information extraction can support the classification of documents into relevant and irrelevant sets, producing results superior to those produced by more standard document classification methods.

Many challenges are currently facing the emerging field of biomedical literature mining. For the sake of effectiveness, broad applicability, and flexibility, it would be useful to reduce the dependency of the mining engines on vocabularies, lexica, or nomenclature. This can be achieved through automated recognition of biomedical entities within text (as demonstrated, for instance, by Fukuda *et al.* [1998] and by Hanisch *et al.* [2003]), or by developing methods that address specific tasks while depending less on explicit terms and more on context (Shatkay *et al.*, 2000).

Explicit low-level immediate tasks involve fact finding, for instance, discovering gene–disease interactions or gene–drug interactions within a vast collection of articles. Another task is the integration of literature into the result analysis of large-scale experiments and high-throughput pipelines, such as homology detection and coexpression analysis. More ambitious goals include the discovery and prediction of metabolic, signaling, or regulatory pathways based on the individual facts reported throughout the literature. Krauthammer *et al.* (2002) provide an interesting discussion with implications to the feasibility and the actual meaning of such a task.

One of the most pressing higher-level needs is the construction of benchmarks and procedures for evaluating the utility of biomedical literature mining tools. Efforts in this direction currently include the recent KDD-cup 2002 competition (Yeh *et al.*, 2002), the new TREC Genomics (Hersh *et al.*, 2003), and the planned BioCreAtIvE forum (Blaschke *et al.*, 2003).

As literature mining challenges in the context of bioinformatics vary widely in aspects such as scope, data sources, and ultimate goals, no single tool can currently perform all the required tasks. However, a combination of methods is likely to address many of the problems. To successfully mine the biomedical literature, it is important to realize the merits and the limitations of the different literature-mining methods. Moreover, it is essential to coherently state the actual biomedical problems we expect to address by using such methods.

ACKNOWLEDGMENTS

We thank the ClearForest-Celera team for their work towards the KDD-cup 2002. HS thanks Stephen Edwards, Mark Boguski, and John Wilbur for their collaboration on the GenTheme project. HS was supported by an NIH IRTA fellowship while developing GenTheme, and was a member of the Informatics Research group at Celera/ABI when writing this article.

REFERENCES

Allen, J. 1995. *Natural Language Understanding*, Benjamin Cummings.

Altschul, S.F., *et al.* 1997. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl. Acids Res.* 25(17), 3389–3402.

Andrade, M.A., and Valencia, A. 1997. Automatic annotation for biological sequences by extraction of keywords from MEDLINE abstracts. Development of a prototype system. *Proc. AAAI Conf. on Intelligent Systems for Molecular Biology (ISMB)*.

Appelt, D.E., and Israel, D.J. 1999. Introduction to information extraction technology. A tutorial prepared for the International Joint Conference on Artificial Intelligence (IJCAI-99). *www.ai.sri.com/~appelt/ie-tutorial/IJCAI99.pdf*.

Aumann, Y., *et al.* 1999. Circle graphs: New visualization tools for text-mining. *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD-97)*, 277–282.

Bader, G.D., *et al.* 2003. BIND: The biomolecular interaction network. *Nucl. Acids Res.* 31(1), 248–250. *www.bind.ca*.

Bafna, V., and Huson, D.H. 2000. The conserved exon method for gene finding. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 3–12.

Bassett, D.E., *et al.* 1999. Gene expression informatics—it's all in your mine. *Nature Genet.* 21, 51–55.

Bear, J., *et al.* 1997. Using information extraction to improve document retrieval. *Proc. 6th Text Retrieval Conf. (TREC-6)*, 367–377.

Ben-Dor, A., *et al.* 1999. Clustering gene expression patterns. *J. Comp. Biol.* 6(3/4), 281–297.

Blaschke, C., *et al.* 1999. Automatic extraction of biological information from scientific text: Protein–protein interactions. *Proc. AAAI Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 60–67.

Blaschke, C., *et al.* 2003. BioCreAtIvE: Critical assessment of information extraction systems in biology. *www.pdg.cnb.uam.es/BioLink/BioCreative.eval.html*.

Blaschke, C., and Valencia, A. 2002. The frame-based module of the SUISEKI information extraction system. *IEEE Intelligent Systems, Special Issue on Intelligent Systems in Biology*, 17(2), 14–20.

Boeckmann, B., *et al.* 2003. The SWISS-PROT Protein Knowledgebase and its supplement TrEMBL in 2003. *Nucl. Acids Res.* 31(1), 365–370. *www.expasy.org/sprot/*.

Brill, E. 1992. A simple rule-based part of speech tagger. *Proc. 3rd Ann. Conf. on Applied Natural Language Processing, ACL*.

Brill, E. 1999. Unsupervised learning of disambiguation rules for part of speech tagging, *in* Armstrong, S. *et al.*, eds., *Natural Language Processing Using Very Large Corpora*, Kluwer Academic, NY.

Burge, C.B., and Karlin, S. 1998. Finding the genes in a genomic DNA. *Current Opin. Struct. Biol.* 8, 346–354.

Cardie, C. 1997. Empirical methods in information extraction. *AI Magazine* 18(4), 65–80.

Chang, J.T., *et al.* 2001. Including biological literature improves homology search. *Proc. Pacific Symposium on Biocomputing (PSB)*, 374–383.

Charniak, E. 1993. *Statistical Language Learning*, MIT Press, New Haven, CT.

Cohen, W.W., and Singer, Y. 1999. Context-sensitive learning methods for text categorization. *ACM Transaction on Information Systems*, 17(2), 141–173.

Cowie, J., and Lehnert, W. 1996. Information extraction. *Communications of the ACM*, 39(1), 80–91.

Craven, M., and Kumlien, J. 1999. Constructing biological knowledge bases by extracting information from text sources. *Proc. AAAI Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 77–86.

Daille, B., *et al.* 1994. Towards automatic extraction of monolingual and bilingual terminology. *Proc. Int. Conf. on Computational Linguistics (COLING-94)*, 515–521.

Darken, R.S., *et al.* 2002. The planar polarity gene strabismus regulates convergent extension movements in xenopus. *European Molecular Biology Organization J. (EMBO)*, 21(5), 976–985.

Deerwester, S., *et al.* 1990. Indexing by latent semantic analysis. *J. Soc. Inf. Sci.* 41(6), 391–407.

DeRisi, J. *et al.* 1997. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* 278, 680–686.

Dermatas, E., and Kokkinakis, G. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics* 21(2), 137–163.

Ding, J., *et al.* 2002. Mining medline: Abstracts, sentences or phrases. *Proc. Pacific Symposium on Biocomputing (PSB)*, 326–337.

Dolinski, K., *et al.* 2000. Saccharomyces genome database. *www.yeastgenome.org/*.

Donaldson, I., *et al.* 2003. PreBind and textomy—Mining the biomedical literature for protein–protein interactions using a support vector machine. *BMC (BioMed Central) Bioinformatics*, 4(11). *www.biomedcentral.com/1471-2105/4/11*.

Dumais, S.T. 1990. Enhancing performance in latent semantic (LSI) indexing. *Behavior Research Methods, Instruments and Computers*, 23(2), 229–236.

Dumais, S.T., *et al.* 1998. Inductive learning algorithms and representations for text categorization. *Proc. 7th Int. Conf. on Information and Knowledge Management (CIKM-98)*, 148–155.

Dumais, S.T., *et al.* 1988. Using latent semantic analysis to improve access to textual information. *Proc. Conf. Human Factors in Computing (CHI88)*.

Elworthy, D. 1994. Does Baum–Welch re-estimation help taggers? *Proc. 4th Conf. Applied Natural Language Processing, ACL*.

Emanuelsson, O., *et al.* 2000. Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *J. Mol. Biol.* 300, 1005–1016.

Feldman, R., *et al.* 2000. A framework for specifying explicit bias for revision of approximate information extraction rules. *Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD-2000)*, 189–197.

Feldman, R., *et al.* 2002. Comparative study of information extraction strategies. *Proc. CICLing-02*, 349–359.

Feldman, R., *et al.* 2001. A domain independent environment for creating information extraction modules. *Proc. Int. Conf. on Information and Knowledge Management (CIKM-01)*, 586–588.

Fisher, D., *et al.* 1995. Description of the UMass systems as used for MUC-6. *Proc. 6th Message Understanding Conference (MUC-6)*, 127–140.

Francis, W.N., and Kucera, H. 1979. Brown Corpus Manual. *www.hit.uib.no/icame/brown/bcm.html*.

Frantzi, T. 1997. Incorporating context information for the extraction of terms. *Proc. ACL-EACL-97*.

Frawley, W.J., *et al.* 1991. Knowledge discovery in databases: An overview, *in* Piatetsky-Shapiro, G., and Frawley, W.J., eds., *Knowledge Discovery in Databases*, 1–27, MIT Press.

Friedman, C., *et al.* 2001. Genies: A natural-language processing system for the extraction of molecular pathways from journal articles. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, S74–S82.

Fukuda, K., *et al.* 1998. Toward information extraction: Identifying protein names from biological papers. *Proc. Pacific Symposium on Biocomputing (PSB)*, 705–716.

Furnas, G.W., *et al.* 1988. Information retrieval using a singular value decomposition model of latent semantic structure. *Proc. Int. ACM Conf. on Research and Development in Information Retrieval (SIGIR-88)*.

Ghanem, M.M., *et al.* 2003. Automatic scientific text classification using local patterns: KDD Cup 2002 (task 1). *SIGKDD Explorations* 4(2), 95–96.

Goldszmidt, M., and Sahami, M. 1998. A probabilistic approach to full-text document clustering. Tech. rep. ITAD-433-MS-98-044, SRI Int.

Greene, B.B., and Rubin, G.M. 1971. Automatic grammatical tagging of English. Tech. rep., Brown University, Providence, RI.

Grishman, R. 1995. The NYU system for MUC-6 or where's the syntax. *Proc. 6th Message Understanding Conference (MUC-6)*, Columbia, Maryland.

Hahn, U., *et al.* 2002. Creating knowledge repositories from biomedical reports: The MEDSYNDICATE text mining system. *Proc. Pacific Symposium on Biocomputing (PSB)*, 338–349.

Hanisch, D., *et al.* 2003. Playing biology's name game: Identifying protein names in scientific text. *Proc. Pacific Symposium on Biocomputing (PSB)*, 403–411.

Hayes, P. 1992. Intelligent high-volume processing using shallow, domain-specific techniques, in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 227–242. Lawrence Erlbaum Assoc., Hillsdale, NJ.

Hayes, P., and Weinstein, S. 1990. CONSTRUE: A system for content-based indexing of a database of news stories. *Proc. 2nd Annual Conf. on Innovative Applications of Artificial Intelligence*.

Hearst, M.A. 1999. Untangling text data mining. *Proc. 37th Annual Meeting of the Association for Computational Linguistics*, 3–10.

Hersh, W., *et al.* 2003. TREC genomics track. *medir.ohsu.edu/∼genomics*.

Hindle, D. 1989. Acquiring disambiguation rules from text. *Proc. 27th Annual Meeting of the Association for Computational Linguistics*.

Hobbs, J. 1986. Resolving pronoun references, in *Readings in Natural Language Processing*, 339–352. Morgan-Kaufmann, Los Altos, CA.

Hobbs, J.R. 1991. SRI International's TACITUS System: MUC-3 test results and analysis. *Proc. 3rd Message Understanding Conference (MUC-3)*, 105–107.

Hobbs, J. R., *et al.* 1991. The TACITUS System Tech. rep. 511, SRI.

Hodges, P.E., *et al.* 1999. The Yeast Proteome Database (YPD): A model for the organization and presentation of genome-wide functional data. *Nucl. Acids Res.* 27(1), 69–73. *www.incyte.com/bioknowledge* (Formerly: *www.proteome.com/YPDhome.html*).

Hofmann, T. 1999. Probabilistic latent semantic indexing. *Proc. 22nd ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-99)*.

Horton, P., and Nakai, K. 1997. Better prediction of protein cellular localization sites with the K nearest neighbors classifier. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*.

HUGO. 2003. HUGO (The Human Genome Organization) Gene Nomenclature Committee. *www.gene.ucl.ac.uk/nomenclature*.

Humphreys, K., *et al.* 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. *Proc. Pacific Symposium on Biocomputing (PSB)*, 502–513.

Iliopoulos, I., *et al.* 2001. TEXTQUEST: Document clustering of medline abstracts for concept discovery in molecular biology. *Proc. Pacific Symposium on Biocomputing (PSB)*, 384–395.

Jaakkola, T., *et al.* 2000. A discriminative framework for detecting remote protein homologies. *J. Comp. Biol.* 7(1/2), 95–114.

Jenssen, T.-K., *et al.* 2001. A literature network of human genes for high-throughput analysis of gene expression. *Nature Genet.* 28, 21–28.

Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. *Proc. European Conf. on Machine Learning (ECML-98).*

KDD. 2002. KDD Cup Competition Web Site. *www.biostat.wisc.edu/∼craven/kddcup*.

Kim, J.D., *et al.* 2003. GENIA Corpus—A semantically annotated corpus for bio-textmining. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)* 180–182. *www-tsujii.is.s.u-tokyo.ac.jp/GENIA*.

Korf, I., *et al.* 2001. Integrating genomic homology into gene structure prediction. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, S140–S148.

Krauthammer, M., *et al.* 2002. Of truth and pathways: Chasing bits of information through myriads of articles. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, S249–S257.

Kupiec, J. 1992. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6.

Lappin, S., and Leass, H.J. 1994. An algorithm for pronominal anaphora resolution. *Computational Linguistics* 20(4), 535–561.

Larkey, L.S., and Croft, W.B. 1996. Combining classifiers in text categorization. *Proc. 19th ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-96)*, 289–297.

Leek, T.R. 1997. Information extraction using hidden Markov models. Master's thesis, Department of Computer Science, University of California, San Diego.

Lehnert, W., *et al.* 1991. Description of the CIRCUS System as used for MUC-3. *Proc. 3rd Message Understanding Conference (MUC-3)*, 223–233.

Lent, B., *et al.* 1997. Discovering trends in text databases. *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining (KDD-97).*

Lewis, D.D. 1995. Evaluating and optimizing autonomous text classification systems. *Proc. 18th ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-95)*, 246–254.

Lewis, D.D., and Hayes, P.J. 1994. Guest editorial for the special issue on text categorization. *ACM Transactions on Information Systems*, 12(3).

Lewis, D.D., and Ringuette, M. 1994. A comparison of two learning algorithms for text categorization. *Proc. 3rd Annual Symposium on Document Analysis and Information Retrieval (SDAIR-94)*, 81–93.

Lewis, D.D., *et al.* 1996. Training algorithms for linear text classifiers. *Proc. 19th ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-96)*, 298–306.

Lewis, D.D. 1997. Test collections: Reuters-21578. *www.daviddlewis.com/resources/testcollections/reuters21578*.

Lindberg, D.A., *et al.* 1993. The unified medical language system. *Meth. Inform. Med.* 32(4), 281–291. *www.nlm.nih.gov/research/umls*.

Lockhart, D.J., *et al.* 1996. Expression monitoring by hybridization to high-density oligonucleotide array. *Nature Biotechnol.* 14, 1675–1680.

Maltese, G., and Mancini, F. 1991. A technique to automatically assign parts-of-speech to words taking into account word-ending information through a probabilistic model. *Proc. of Eurospeech-91*, 753–756.

Marcotte, E.M., *et al.* 2001. Mining literature for protein-protein interactions. *Bioinformatics* 17(4), 359–363.

Marcus, M. 1992. The Penn Treebank Project. *www.cis.upenn.edu/∼treebank*.

Merialdo, B. 1994. Tagging english text with a probabilistic model. *Computational Linguistics* 22(2), 155–172.

Mitkov, R. 1998. Robust pronoun resolution with limited knowledge. *COLING-ACL*, 869–875.

Myers, E. 1999. Whole-genome DNA sequencing. *IEEE Computational Engineering and Science* 3(1), 33–43.

NIST. 1987–1998. Message understanding conference (MUC). *www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html*.

NLM. 2003. Mesh: Medical subject headings. *www.nlm.nih.gov/mesh/*.

OMIM. 2000. Online Mendelian inheritance in man. *www.ncbi.nlm.nih.gov/omim/*.

Park, J.C., *et al.* 2001. Bidirectional incremental parsing for automatic pathway identification with combinatory categorical grammar. *Proc. Pacific Symposium on Biocomputing (PSB)*, 396–407.

Pearson, H. 2001. Biology's name game. *Nature* 411, 631–632.

Ponte, J.M., and Croft, W.B. 1998. A language modeling approach to information retrieval. *Proc. 21st ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-98).*

Porter, M.F. 1997. An algorithm for suffix stripping (reprint), in *Readings in Information Retrieval*, Morgan Kaufmann. *www.tartarus.org/∼martin/PorterStemmer/.*

Pruitt, K.D., and Maglott, D.R. 2001. RefSeq and LocusLink: NCBI gene-centered resources. *Nucl. Acids Res.* 29(1), 137–140. *www.ncbi.nlm.nih.gov/LocusLink*.

Pustejovsky, J., *et al.* 2002. Robust relational parsing over biomedical literature: Extracting inhibit relations. *Proc. Pacific Symposium on Biocomputing (PSB)*, 362–373.

Rajman, M., and Besançon, R. 1997. Text mining: Natural language techniques and text mining applications. *Proc. 7th IFIP 2.6 Working Conf. on Database Semantics, DS-7*.

Ray, S., and Craven, M. 2001. Representing sentence structure in hidden Markov models for information extraction. *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI-01).*

Regev, Y., *et al.* 2002. Rule-based extraction of experimental evidence in the biomedical domain—the KDD Cup 2002 (task 1). *SIGKDD Explorations* 4(2), 90–92.

Renner, A., and Aszódi, A. 2000. High-throughput functional annotation of novel gene products using document clustering. *Proc. Pacific Symposium on Biocomputing (PSB)*.

Riloff, E., and Lehnert, W. 1994. Information extraction as a basis for high-precision text classification. *ACM Transactions on Information Systems* 12(3), 296–333.

Rindflesch, T.C., *et al.* 2000. Edgar: Extraction of drugs, genes and relations from the biomedical literature. *Proc. Pacific Symposium on Biocomputing (PSB)*, 514–525.

Russell, S.J., and Norvig, P. 1995. *Artificial Intelligence, A Modern Approach*, chap. 22–23, Prentice Hall, Englewood Cliffs, NJ.

Sahami, M. 1998. *Using Machine Learning to Improve Information Access*. Ph.D. thesis, Stanford University, Computer Science Department.

Sahami, M., *et al.* 1996. Applying the multiple cause mixture model to text categorization. *Proc. 13th Int. Conf. on Machine Learning*.

Salton, G. 1989. *Automatic Text Processing*, Addison-Wesley, Reading, MA.

Schena, M., *et al.* 1995. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science* 270, 467–470.

Schütze, H. 1993. Part-of-speech induction from scratch. *Proc. 31st Annual Meeting of the Association for Computational Linguistics*, 251–258.

Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Computing Surveys* 34(1), 1–47.

Sharan, R., and Shamir, R. 2000. CLICK: A clustering algorithm with applications to gene expression analysis. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 307–316.

Shatkay, H., *et al.* 2002. Information retrieval meets gene analysis. *IEEE Intelligent Systems, Special Issue on Intelligent Systems in Biology* 17(2), 45–53.

Shatkay, H., *et al.* 2000. Genes, themes and microarrays: Using information retrieval for large scale gene analysis. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*, 317–328.

Shatkay, H., and Wilbur, W.J. 2000. Finding themes in medline documents: Probabilistic similarity search. *Proc. IEEE Conf. on Advances in Digital Libraries*, 183–192.

Shaw, W.M., *et al.* 1997. Performance standards and evaluations in IR test collections: Cluster-based retrieval models. *Inf. Proc. Management* 33(1), 1–14.

Sonnhammer, E.L.L., *et al.* 1998. A hidden Markov model for predicting transmembrane helices in protein sequences. *Proc. Int. Conf. on Intelligent Systems for Molecular Biology (ISMB)*.

Spellman, P.T., *et al.* 1998. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces Cerevisiae* by microarray hybridization. *Molecular Biology of the Cell* 9, 3273–3297.

Stapley, B.J., and Benoit, G. 2000. Bibliometrics: Information retrieval and visualization from co-occurrences of gene names in medline abstracts. *Proc. Pacific Symposium on Biocomputing (PSB)*, 526–537.

Stapley, B.J., *et al.* 2002. Predicting the subcellular location of proteins from text using support vector machines. *Proc. Pacific Symposium on Biocomputing (PSB)*, 374–385.

Stephens, M., *et al.* 2001. Detecting gene relations from medline abstracts. *Proc. Pacific Symposium on Biocomputing (PSB)*, 483–496.

Swanson, D.R. 1986. Fish-oil, Raynaud's syndrome and undiscovered public knowledge. *Perspectives in Biology and Medicine* 30(1), 7–18.

Swanson, D.R. 1988. Migraine and magnesium: Eleven neglected connections. *Perspectives in Biology and Medicine* 31(4), 526–557.

Swanson, D.R. 1990. Somatomedin C and arginine: Implicit connections between mutually isolated literatures. *Perspectives in Biology and Medicine* 33(2), 157–186.

Tamayo, P., *et al.* 1999. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. of Sci.* 96, 2907–2912.

Tanabe, L., *et al.* 1999. MedMiner: An internet text-mining tool for biomedical information, with application to gene expression profiling. *BioTechniques* 27(6), 1210–1217.

The FlyBase Consortium. 2002. The FlyBase database of the drosophila genome projects and community literature. *Nucl. Acids Res.* 30(1), 106–108. *www.flybase.org*.

The Gene Ontology Consortium. 2000. Gene ontology: Tool for the unification of biology. *Nature Genet.* 25, 25–29. *www.geneontology.org*.

Thomas, J., *et al.* 2000. Automatic extraction of protein interactions from scientific abstracts. *Proc. Pacific Symposium on Biocomputing (PSB)*, 538–549.

van Rijsbergen, C.J. 1977. A theoretical basis for the use of co-occurrence data in information retrieval. *J. Documentation* 33(2), 106–119.

van Rijsbergen, C.J. 1979. *Information Retrieval*, Butterworth, London.

Vapnik, V. 1995. *The Nature of Statistical Learning Theory*, Springer-Verlag, NY.
Venter, J.C., *et al.* 2001. The sequence of the human genome. *Science* 291, 1304–1351.
Voorhees, E., and Harman, D.K. 1993. Text REtrieval Conference (TREC). *trec.nist.gov*.
Weischedel, R., *et al.* 1993. Coping with ambiguity and unknown words through probabilistic methods. *Computational Linguistics*, 19.
Wilbur, W.J. 1992. An information measure of retrieval performance. *Information Systems* 17(4), 283–298.
Wilbur, W.J., and Coffee, L. 1994. The effectiveness of document neighboring in search enhancement. *Information Processing and Management* 30(2), 253–266.
Wilbur, W.J., and Yang, Y. 1996. An analysis of statistical term strength and its use in the indexing and retrieval of molecular biology text. *Computers in Biology and Medicine* 26(3), 209–222.
Witten, I.H., *et al.* 1999. *Managing Gigabytes, Compressing and Indexing Documents and Images* (2nd ed.), Morgan-Kaufmann, San Diego, CA.
Yakushiji, A., *et al.* 2001. Event extraction from biomedical papers using a full parser. *Proc. Pacific Symposium on Biocomputing (PSB)*, 408–419.
Yandell, M.D., and Majoros, W.H. 2002. Genomics and natural language processing. *Nature Reviews* 3, 601–610.
Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 69–90.
Yang, Y., and Chute, C.G. 1994. An example-based mapping method for text categorization and retrieval. *ACM Trans. Inf. Systems* 12(3), 252–277.
Yang, Y., and Liu, X. 1999. A re-examination of text categorization methods. *Proc. 22nd ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR-99)*, 42–49.
Yeh, A., *et al.* 2002. Background and overview for KDD Cup 2002 task 1: Information extraction from biomedical articles. *SIGKDD Explorations* 4(2), 87–89.

Address correspondence to:
Hagit Shatkay
School of Computing
Queen's University
Kingston, Ontario
Canada K7L3N6

E-mail: shatkay@cs.queensu.ca