Journal of Proteomics & Bioinformatics - Open Access

Research Article JPB/Vol.2/August 2009

GeneNarrator: Mining the Literaturome for Relations Among Genes

Jing Ding[1], Daniel Berleant[2*], Jun Xu[3], Kenton Juhlin[4], Eve Wurtele[5], Andy Fulmer[6]

[1]Information Warehouse, Ohio State University Medical Center, 410 W. 10th Ave., Columbus, Ohio, 43210, USA, jing.ding@osumc.edu, (614) 293-0776, fax (614) 293-2210

[2]Department of Information Science, University of Arkansas at Little Rock, 2801 University Ave., Little Rock, Arkansas, 72204, USA, berleant@gmail.com, (501) 683-7056, fax (501) 683-7049

[3]Miami Valley Laboratories, The Procter and Gamble Company, 11810 East Miami River Rd., Ross, Ohio, 45061, USA, xu.j.1@pg.com

[4]Miami Valley Laboratories, The Procter and Gamble Company, 11810 East Miami River Rd., Ross, Ohio, 45061, USA, juhlin.kd@pg.com

[5]Department of Genetics, Development and Cell Biology, Iowa State University, Ames, Iowa, 50011, USA, mash@iastate.edu

[6]Miami Valley Laboratories, The Procter and Gamble Company, 11810 East Miami River Rd., Ross, Ohio, 45061, USA, fulmer.aw@pg.com

*Corresponding author: Daniel Berleant, Department of Information Science, University of Arkansas at Little Rock, 2801 University Ave., Little Rock, Arkansas, 72204, USA, E-mail:berleant@gmail.com; Tel: (501) 683-7056; Fax: (501) 683-7049

Received July 07, 2009; Accepted August 23, 2009; Published August 24, 2009

Citation: Ding J, Berleant D, Xu J, Juhlin K, et al. (2009) GeneNarrator: Mining the Literaturome for Relations Among Genes. J Proteomics Bioinform 2: 360-371. doi:10.4172/jpb.1000096

Copyright: © 2009 Ding J, et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract

The rapid development of microarray and other genomic technologies now enables biologists to monitor the expression of hundreds, even thousands of genes in a single experiment. Interpreting the biological meaning of the expression patterns still relies largely on biologist's domain knowledge, as well as on information collected from the literature and various public databases. Yet individual experts' domain knowledge is insufficient for large data sets, and collecting and analyzing this information manually from the literature and/or public databases is tedious and time-consuming. Computer-aided functional analysis tools are therefore highly desirable.

We describe the architecture of GeneNarrator, a text mining system for functional analysis of microarray data. This system's primary purpose is to test the feasibility of a more general system architecture based on a two-stage clustering strategy that is explained in detail. Given a list of genes, GeneNarrator collects abstracts about them from PubMed, then clusters the abstracts into functional topics in a first clustering stage. In the second clustering stage, the genes are clustered into groups based on similarities in their distributions of occurrence across topics. This novel two-stage architecture, the primary contribution of this project, has benefits not easily provided by one-stage clustering.

Keywords: Genes; Clustering; Text mining

Introduction

Rapid developments in genomic technologies such as microarrays now enable biologists to simultaneously monitor the expression of thousands of genes in a single experiment. Automated data analysis methods and software tools are important for efficiently processing the resulting large amounts of data. Numerous algorithms and tools have been developed for finding patterns in gene expression and grouping genes with similar patterns.

Interpreting the biological meanings of the patterns, however, still largely relies on human experts' domain knowledge, as well as on manual collection of previously reported results. Human expertise works well when experts are available, as for specific domains and their relatively modestly sized data sets. However, human expertise can be expensive. It is also not always available. For example systems biology research often depends on interdisciplinary collaborations across multiple domains.

That is in the nature of large, complex systems. Some of the researchers might not be experts in even one domain, students are often involved who are not domain experts, and in any case it is unrealistic to expect even domain experts to memorize functional details of thousands of genes. The alternative of manually collecting and analyzing them from the literature and public databases is tedious and time-consuming. Therefore, computer-aided functional analysis tools are a critical need. To help meet this need, this work contributes a novel two-stage clustering architecture for which, using a test implementation called GeneNarrator, we demonstrate the feasibility.

Related Work and Motivation

Numerous system architectures for functional analysis of microarray data, and the systems that demonstrate them, have been reported in the literature. In terms of the sources of the functional information they rely on, they can be grouped into two categories.

- Architectures that rely on curated lexicons, ontologies, and other such functional annotation sources, such as the Gene Ontology (Adryan and Schuh, 2004; Badea 2003; Joslyn et al., 2004; Kennedy et al., 2004; Pasquier et al., 2004; Robinson et al., 2004; Smid and Dorssers, 2004; Khatri and Draghici, 2005; Masys et al., 2001; and Kankar et al., 2002). Another well-know curated resource is Reactome (http://www.reactome.org), and there are others as well.

- Architectures that derive functional information from MEDLINE and other text resources (Becker et al., 2003, Chaussabel and Sher, 2002; Homayouni et al., 2005; Kim and Falkow, 2003; Oliveros et al., 2000, Raychaudhuri and Altman, 2003; Raychaudhuri et al., 2002, Renner and Aszodi, 2000; Slaton and McGill, 1983; Glenisson et al., 2003; Chagoyen et al., 2006; Shatkay et al., 2000; Mao et al., 2005).

These categories are further discussed next.

Using GO, a representative curated resource

The Gene Ontology (GO) is a controlled vocabulary for describing genes. Its development by a group of member organizations started in 1998, and is coordinated by the Gene Ontology Consortium. The GO Consortium also acts as a repository of gene and gene product annotations contributed by member organizations, e.g., FlyBase, the *Saccharomyces* Genome Database (SGD) and the Mouse Genome Database (MGD). This makes it a valuable source of functional information for annotating microarray experiments, and various groups have explored strategies for using this information to functionally summarize gene clusters. A review of ontologies for functional annotation is provided by Khatri and Draghici, (2005).

Not all works in this category focus on GO. For example Masys et al., (2001); Kankar et al., (2002) used MeSH terms, the former also using EC numbers. The above-mentioned works shared some limitations that result from attributes of GO and are also common in other curated data sources.

- The GO itself, exemplifying the types of problems common for such resources, is not static and mature. As one consequence, its development is unbalanced. While some branches are deep (up to 18 levels) and contain very detailed concepts (e.g., GO:0000201 – nuclear translocation of MAPK during cell wall biogenesis), the GO coverage in some areas of biology is incomplete, for example in pathways (Mao et al., 2005) and immunology (Joslyn et al., 2004).

- The GO is updated regularly, so keeping annotations constantly compatible with the latest GO release is a challenge. And, even the latest release of such a resource may not be up to date as there is a time lag between when new data is available in the literature and when it is annotated and placed into databases.

- GO annotations are mainly curated manually so, as with other human-curated information sources, inconsistencies are endemic. Thus for example Badea, (2003) had to correct numerous mistakes in the Proteome HumanPSD database by hand before performing the analysis.

- GO annotations are mainly available for well-studied genes in a few model organisms. Badea, (2003) could find annotations for only 26% (39 out of 149) of the genes of interest. Analysis based on such incomplete data can be risky.

Text-based functional analysis

One approach to overcoming limitations of systems relying on controlled ontologies is to extract functional information from the texts in online literature databases such as MEDLINE and its PubMed portal (http://www.ncbi.nlm.nih.gov/pubmed/). Other keyword-searchable literature resources like CiteXplore (http://www.ebi.ac.uk/citexplore/) could also be used. Text analysis can be deep or shallow. For example, the deep part of the spectrum includes parsing, template matching, and inference of causal relationships among biomolecules based on the properties of individual sentences.

Shallow analysis can lead to sophisticated annotation systems that facilitate navigating in the biomedical literature by adding hyperlinks and related tools, like Whatizit (http://www.ebi.ac.uk/webservices/whatizit/info.jsf), iHOP (http://www.ihop-net.org/), and WikiHyperGlossary (http://bioinformatics.ualr.edu/HyperGlossary/hg/url). Shallow analysis can also lead to fine-grained retrieval of specific parts of documents, as with MedMiner, a tool that retrieves individual sentences from MEDLINE (Tanabe et al., 1999). But shallow methods can also feed into deeper analyses of text collections. Often this starts with viewing texts as collections of words. Such methods are called *bag of words* (BOW) approaches.

Texts are typically so complex and unstructured that deep analysis can be cumbersome, especially for large quantities of text, as well as error-prone. These problems will likely remain until the grail of full natural language understanding is finally reached at some unknown future time. Interpretation errors will tend to propagate through subsequent inferences, negatively impacting results. Therefore it is useful for text mining system architectures to robustly resist the effects of such text interpretation "noise," and shallow analysis is one strategy.

On the other hand, there are limitations to shallow analysis methods as well. An obvious limitation is that the rich information content in a text is not fully harvested. Another limitation is that the reliability of one text may be much greater than that of another, because the quality of some publications is much greater than that of others, and this is hard to determine automatically. The least reliable texts tend to be less likely to be included in major corpora, which helps but does not fully address the problem. A third limitation is that texts in different languages are difficult to match with one another because translation is needed first. Although potentially feasible, a translation preprocessing step is not yet a component of most text mining systems.

Bag-of-words based text analysis

Text-based functional analysis systems based on the shallow bag-of-words paradigm can be divided into two subcategories, those that make the assumption that genes with similar expression patterns are involved in the same functional pathways, and those that do not. The assumption suggests that MEDLINE abstracts referring to a gene in a cluster of genes with similar expression patterns have significant properties in common that provide hints about the cluster's functional properties. Oliveros et al., (2000) for example focused on the words in MEDLINE abstracts. The significance of a word to a particular cluster of similarly expressed genes was determined by a *z*-test against its average frequency in abstracts relevant to any gene cluster.

However, the assumption that similar expression patterns necessarily mean genes are functionally related is problematic. For example, genes involved in different pathways may have similar expression patterns in a particular experiment, and a single gene may participate in several pathways. In either case, the genes in a cluster would tend to represent different pathways, making their interpretation more difficult. This motivates extracting functional similarities without the assumption. That in turn means clustering based on something other than gene expression profile.

Chagoyen et al., (2006) used non-negative matrix factorization (NMF) to process sets of abstracts related to specific genes. The result was a vector characterizing a given gene in terms of "semantic features," which are weighted, semantically relevant terms extracted automatically from the texts. The vectors of different genes can then be compared in order to cluster genes into "functionally coherent" sets. A limitation of this approach is that the abstracts associated with a gene and processed into a vector must be provided to the system.

Shatkay et al., (2000) developed a theme extraction system for large-scale gene analysis. A theme was derived from a set of MEDLINE abstracts with similar term distributions. The abstract set was obtained from a user-provided kernel document known to be about a particular gene, but from that kernel the system obtained similar documents from MEDLINE automatically, using a standard cosine-based similarity metric. The resulting set was then processed to extract "executive summary" terms defining the theme of the abstract set and, hence, the gene behind it. Theme similarity between two genes was based on the members shared by the themes of their two corresponding abstract sets. The final results included, for each gene, a theme consisting of characteristic terms and a list of the most similar genes. A limitation of the system is its dependence on the kernel documents, which the user must provide. For microarray experiments involving a large number of genes, the time and effort required to identify kernel documents might be prohibitive even when good kernels exist.

The "literature profiling" system of Chaussabel and Sher, (2002) also found gene clusters based on text clustering. The system represented genes as vectors of keywords extracted from MEDLINE abstracts. The vectors were then clustered using a software package originally developed

for gene-expression profiling. The resulting clustergram showed gene clusters with similar keyword profiles. The genes in a cluster were interpreted as functionally related. This text clustering-based approach avoids a significant limitation of the approach of Shatkay et al., (2000); Chagoyen et al., (2006) in that the burden of providing hand-selected documents is avoided (although they mention that providing PubMed queries that produce good document sets can be non-trivial). However, literature profiling has its own limitations.

- The algorithm was originally developed for gene expression clustering, not keyword clustering, and performed poorly on high-dimensional vectors. The authors were forced to filter the keywords aggressively to reduce the dimensionality, retaining just 101 out of 25,000 terms, a significant loss of information.
- The clusters were dominated by well-studied genes with rich keyword profiles, because they were mentioned in many abstracts. Newly discovered or less-studied genes had only a few abstracts and so were relatively neglected because of their sparsely populated profiles.
- The system only counted unigrams. Thus functional information in multiple-word terms was left unused. For example, the meaning of "red blood cell" is difficult to capture from the separate words "red," "cell" and "blood" scattered among many other keywords.

Although the Chagoyen et al., (2006); Shatkay et al., (2000) Chaussabel, (2002) and Sher, (2002) systems avoided the inherent short- comings of assuming that genes with similar expression patterns are involved in the same functional pathways, they still required facing the significant challenges of effectively extracting, processing and presenting functional information from free text.

Glenisson et al., (2003) addressed this issue with an approach similar to the literature profiling of Chaussabel and Sher, (2002) but with the following significant differences.

- Each gene's functional description was collected from the Saccharomyces Genome Database and the SWISS-PROT database, supplemented with 20 MEDLINE abstracts. But this requires data availability, limiting the analysis to well-studied genes.
- The functional descriptions were represented in a predefined vector space of GO concepts.

Because of its reliance on GO, limitations of GO as discussed above apply to this system, e.g., its unbalanced development and incomplete coverage. This tends to counteract the objective of using the literature to obtain the most comprehensive and up-to-date functional information.

GeneNarrator

We created a new system architecture which addressed limitations of the above-mentioned tools with an approach based on two clustering stages. The feasibility of this architecture was tested by building an example system, GeneNarrator, based on it. There are five requirements which the architecture meets.

1. Intended application

The architecture is intended for functional interpretation of microarray experiments for which information exists in the literature about the genes involved. However, with modest modification, it should be adaptable to proteomic and metabolomic data as well.

2. Input

Input should be as simple as possible — for example, a list of gene names.

3. Source of functional information

Functional information should be obtained from MEDLINE abstracts, since they are fairly comprehensive and up-to-date. Functional annotations in public genomic databases tend to cover well-studied genes in model organisms, so depending on them would have made the architecture inapplicable to less-studied genes or non-model organisms. Avoiding reliance on these sources also avoids concerns about annotation errors and update delays.

4. Algorithmic constraints

The analysis should provide a summarized picture of the thousands or even tens of thousands of MEDLINE abstracts that may be collected for a given list of genes. Text clustering is a natural choice, because it can divide a large number of documents into groups based on topic differences (usually based on similarities in word content). The clustering solution should be designed to avoid the pitfalls illustrated in some of the above-reviewed systems.

- Well-studied genes with many abstracts should not dominate the clustering process to the relative neglect of newly discovered or less-studied genes.
- Hierarchical clustering algorithms are more suitable than flat ones. Users rarely know beforehand how

many clusters should be in the final result. With hierarchical clustering, the number of clusters is relatively flexible, as branches can be merged after the analysis.

- The text-clustering algorithm of choice should perform well in high-dimensional vector spaces.
- Multiple-word terms should be incorporated in text clustering. Many biomedical concepts are multiword terms. Breaking them down to unrelated single words may adversely affect clustering results, because useful information is lost.

5. Output of results

Results of an analysis should include a hierarchical structuring of topics, the biological meanings of the topics, and how many and what genes are in what topics.

The Algorithms, Architecture, and Implementation

Overview

A two-stage clustering approach was designed for GeneNarrator to use to provide functional summarizations of microarray experiments from information in MEDLINE (Fig. 1). The system is designed to take as input a user-provided gene list, and automatically queries PubMed for abstracts mentioning one or more of the genes. Gene symbols, official names, synonyms and gene product names could potentially all be included to retrieve more relevant abstracts. PubMed uses a sophisticated query expansion method to increase the recall of relevant records. However, increased recall in general tends to reduce precision. This is an issue with most systems that use information retrieval rather than hand-curated data sets as input. An example of this specific to the gene annotation problem is ambiguous gene names, which tend to occur most in the most widely studied organisms. The precision will depend heavily on the particular gene list.

The system is designed to group the pool of retrieved abstracts into functional *topics* using a text clustering algorithm. Next, each gene is associated with the distribution of its occurrences across the set of topics, i.e., a vector stating its number of occurrences in the MEDLINE records comprising each topic. Then, a second clustering stage groups genes with similar distributions.

GeneNarrator, the name of the demonstration implementation, consists of six modules: DocBuilder, LongBOW (BOW from "Bag Of Words"), CrossBOW, GeneSmith, ArrowSmith (not related to the Arrowsmith system, http://arrowsmith.psych.uic.edu/arrowsmith_uic/index.html), and BOWviewer (Fig. 2). DocBuilder retrieves MEDLINE abstracts that are related to at least one of the user-provided genes. LongBOW performs several preprocessing tasks on the abstracts, including discarding stopwords, stemming, and detecting multiple-word terms. CrossBOW clusters the abstracts into a hierarchy of functional topics. ArrowSmith extracts representative keywords from CrossBOW's output, and scores keyword-containing sentences and abstracts. The keywords and high-scoring sentences and abstracts are intended to help in interpreting the biological meanings of the topics. GeneSmith calculates, for each gene, a distribution describing its occurrences

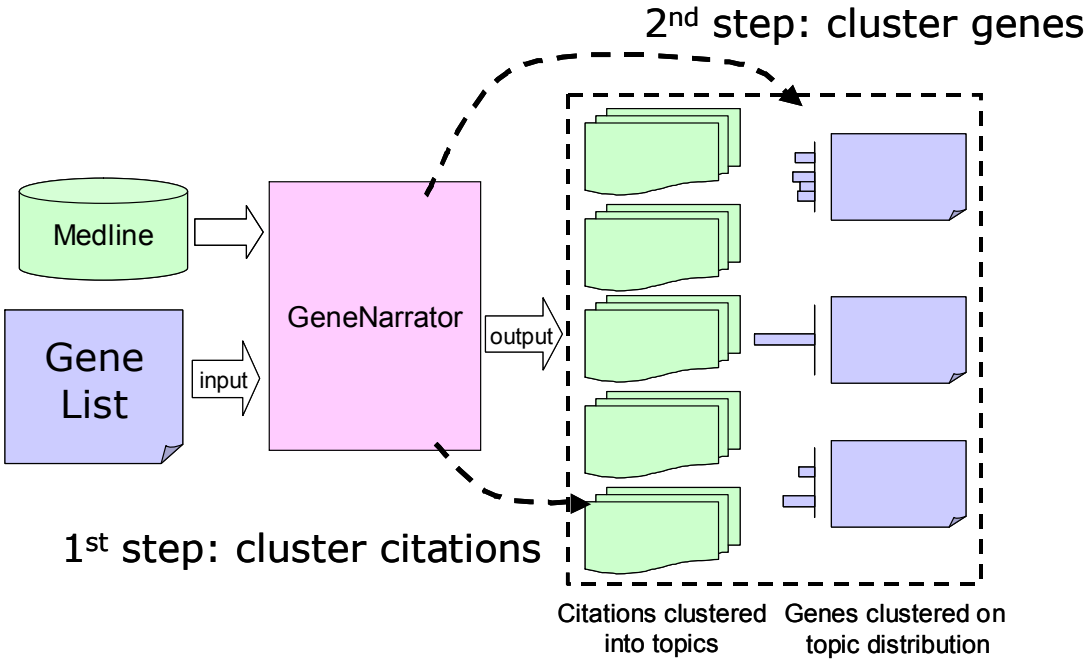

Figure 1: Functional overview of GeneNarrator.

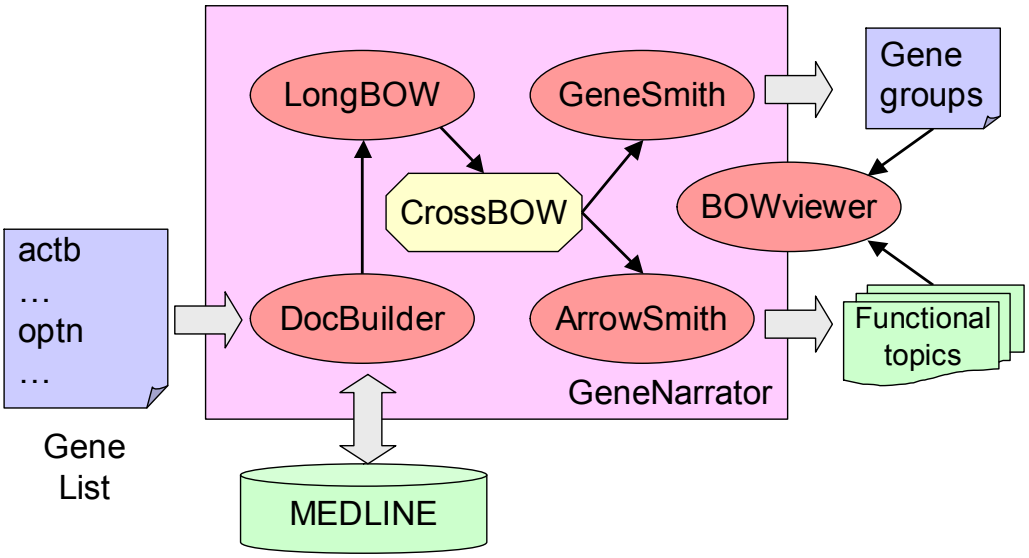

Figure 2: Architectural overview of GeneNarrator.

across the topics, then clusters the genes with similar distributions. BOWviewer is a GUI for navigating the hierarchical topics, browsing the representative keywords, sentences and abstracts, and comparing the topic distributions of individual genes or gene clusters. All modules were implemented in Java, except CrossBOW which is in C.

Details

Here we give additional details about the design of the various modules of GeneNarrator, as it may be of interest to future system builders.

Document retrieval (DocBuilder)

Given a file containing a list of genes, one gene per line, the DocBuilder module retrieves MEDLINE abstracts related to each of the genes via eUtils, which are the Entrez Programming Utilities (National Library of Medicine 2004). DocBuilder is designed to hold four submodules, the *querier*, *sampler*, *fetcher* and *parser*. The *querier* embeds a gene name, together with its synonyms and gene product names if provided, into a query which it sends to PubMed using the eUtils ESearch function. PubMed returns a list of PMIDs. A user may set an upper limit for the number of PMIDs to be used in the subsequent processing steps. If the number of returned PMIDs exceeds the upper limit, the *sampler* draws a random sample from the list. The returned or sampled PMIDs are recorded in a gene-to-PMID map file for later use. The *fetcher* then retrieves the PMIDs' full abstracts from PubMed using the eUtils EFetch function. Finally, the *parser* extracts the titles and the abstracts from the retrieved abstracts, and writes them to plain text files. The submodules we built for our demonstration system, GeneNarrator, were used earlier in MedKit (Ding and Berleant, 2005), and PubMed Assistant (Ding et al., 2006).

Preprocessing (LongBOW)

The LongBOW module preprocesses the MEDLINE abstracts in order to get better clustering results. The preprocessing was designed to perform the following steps.

- Remove stop words, such as "that," "is," "you," and "of." A user-defined stopword list can replace the default stopword list.
- Perform stemming. For example "regulation," "regulating," "regulator," and "regulates" are all stemmed to "regulat." The stemming method is a Java implementation (http://www.tartarus.org/martin/PorterStemmer/) of the Porter, (1980) stemming algorithm.
- Detect and label multiple-word terms (MWTs).

Detecting and labeling MWTs is done in three passes through the abstracts. The first pass performs stopword removal and stemming. Also, for each unique single-word term (SWT), its *df* (document frequency, or number of documents containing the term) and *tf* (term frequency, or total number of appearances of the term in the entire document set) are counted. The total number of tokens (words, including stopwords, and punctuation marks) is also recorded. After the first pass, a threshold value is used to separate out SWTs that are significant, defined as having a *df* above the threshold.

In the second pass, unique double-word terms (DWTs) are counted. A DWT is defined as two consecutive SWTs

without any intervening stopwords or punctuation. Both constituent SWTs must be above the *df* threshold. A DWT's observed count is tested against the null hypothesis that two SWTs are next to each other by chance. The test is similar to the "*t*-test" of collocations described by Manning and Schultze (1999). Briefly, let the number of occurrences of single-word term w_i in the entire document (in this case, abstract) be n_i, the total number of tokens (words and punctuation marks) in the document set be N, and the number of occurrences of double-word term w_1w_2 be n_{12}. Then the probability of an occurrence of w_1 being followed by w_2 under the null hypothesis is $p=n_2/N$, and the expected number of occurrences of w_{12} is $n_{exp}=n_1p$. We can construct an approximate binomial test by $z=\frac{n_{12}-n_{exp}}{\sqrt{n_1p(1-p)}}$ and use tables for z from many standard statistics texts to decide whether or not to reject the null hypothesis. DWTs for which the null hypothesis is rejected are significant.

In the third pass, significant DWTs are evaluated in the context of individual MEDLINE abstracts if their two constituent SWTs are mentioned a similar number of times in the abstract. For example, an occurrence of "*cell cycle*" would not be used as a DWT in a abstract that mentioned the word "*cell*" 10 times, but the word "*cycle*" only once. Even though "cell cycle" might appear more frequently than expected by chance in the entire abstract set, it would not be deemed a major subject in that abstract. Formally, given a significant DWT w_1w_2 with corresponding constituent SWT term frequencies tf_1 and tf_2 in a abstract, and a predefined threshold $\alpha > 1$, the DWT is used if and only if $1/\alpha \le tf_1/tf_2 \le \alpha$, where α is the sensitivity. Too high a sensitivity will tend to lead too high a vector space dimensionality, which will likely decrease the quality of the clustering for typical data sets. This study used an α value of 3.0.

Finally, MWTs are detected if DWTs chain together. Upon detecting a qualified DWT, LongBOW replaces the space with an underscore character (e.g., *cell_cycle*). This trick enables CrossBOW to treat the DWTs and MWTs as single words.

Text clustering (CrossBOW)

The CrossBOW module was modified from the open source "Bow" toolkit (McCallum, 1996). Its clustering algorithm, the Cluster-Abstraction Model (CAM) (Hofmann, 1999), was designed specifically for text clustering. A CAM consists of a vocabulary, and many topics that are automatically extracted and organized as nodes in a hierarchical tree. Each topic is defined by a vector of probabilities P_t. Each probability in the vector is the likelihood of the topic containing a certain word from the vocabulary. Each leaf topic has a unique path to the root of the tree. The topics along the path are considered to be different abstraction levels. The closer to the root, the higher the abstraction level. There is a document bin for each route. This bin, like a topic, is also associated with a vector of probabilities P_b. Each probability is the likelihood of the bin producing documents from a certain abstraction level. Finally, the entire model has a vector of probabilities P_m, each giving the likelihood of the model producing documents from a certain bin.

Given a model, a set of documents can be generated by iteratively picking a bin according to P_m, picking an abstraction level (that is, a topic) according to P_b given the bin, and producing words according to P_t given the topic. Clustering a set of documents is equivalent to finding the hierarchical topic structure and associated probabilities with the maximum likelihood of generating that set of documents. Compared to other distance-based algorithms, especially agglomerative clustering methods, CAM has the following advantages (Rose et al., 1990):

- insensitivity to term-weighting methods and distance (similarity) definitions,
- a statistically sound foundation,
- multiple levels of text clustering,
- representative keywords for topics, and
- efficient model fitting by annealed expectation maximization.

CrossBOW clusters a set of documents (in plain text format) into a hierarchical topic tree. Each document is assigned to one and only one of the nodes (topics). The branching factor and maximum depth of the tree are designed as command line options. The modifications to CrossBOW introduced for GeneNarrator include:

- recognition of multiword terms labeled by LongBOW, and
- addition of a command line option to change the number of topic keywords in the output.

Interpreting the topic's biological meanings (ArrowSmith)

Given the hierarchical topics and the representative topic keywords generated by CrossBOW, the ArrowSmith mod-

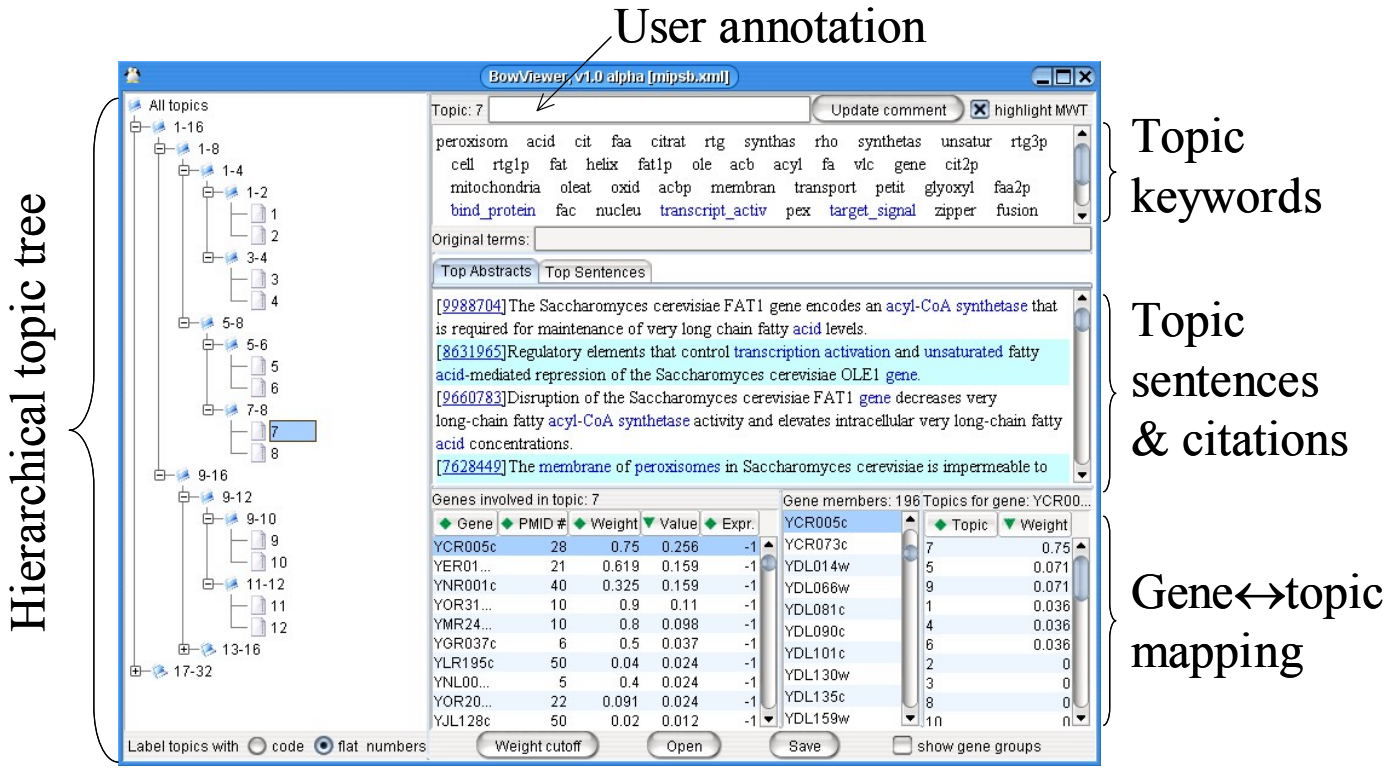

Figure 3: BOWviewer user interface.

ule is designed to score the sentences and the abstracts containing them. Each topic-representative keyword is assigned a keyword score. For example, the binary scoring method gives all representative keywords a score of one. Other scoring methods could assign different scores to different keywords based on their probabilities or ranks. A sentence's score is the sum of its keyword's scores, and an abstract's score is the sum of its sentence's scores. The representative keywords and the highest-scoring sentences and abstracts define the inferred biological meaning of each topic.

Gene-to-topic mapping and clustering (GeneSmith)

The GeneSmith module is designed to convert the abstract set associated with a particular gene into a topic distribution by straightforwardly counting how many abstracts in the set fall into each topic. It then clusters genes based on the similarities of their distributions across topics. The clustering algorithm may be chosen as either *k*-means or expectation maximization (EM) from the Weka machine-learning workbench (Frank et al., 2004).

Result browsing (BOWviewer)

The BOWviewer module (Fig. 3) is a graphical user interface built for browsing the results. It shows the topic hierarchy (left), the representative keywords (top), high-scoring sentences or abstracts for each topic (middle), and other information. Users can readily navigate through the hierarchical topic tree. They can browse topic keywords, high-scoring sentences, and abstracts, and they can annotate the topics with biologically meaningful comments. Users can also check the genes associated with a particular topic (e.g. topic 7 in Fig. 3., lower middle), and the topics and their strengths (lower right corner) associated with a highlighted gene in the gene list (lower middle right).

Results

To validate the architecture we performed an experiment on a list of 155 yeast genes (Table 1) manually selected from ten pathways in the comprehensive yeast genome database (Munich Information Center for Protein Sequences, 2006). Care was taken in picking the pathways so that the overlap was low, though some overlap was unavoidable. The modules were run with the following command line parameters.

DocBuilder:

- maximum number of abstracts per gene = 50

LongBOW:

- single word term *df* (document frequency) threshold = 0.05

- double word term *p* value = 0.025
- double word term *tf* (term frequency) ratio = 3.0
- default stopword lists

CrossBOW:

- branching factor = 2
- maximum branch depth = 4
- number of output keywords per topic = 50

ArrowSmith:

- scoring method = binary
- number of top-scoring sentences and abstracts = 25

GeneSmith:

- clustering algorithm = *k*-means
- *k* = 15

DocBuilder retrieved 2,819 abstracts from MEDLINE, some of which covered two or more genes in the list. CrossBOW was used to generate a topic hierarchy organized as a binary tree. It then assigned each abstract to one of the 32 leaf nodes. Each node was associated with 50 keywords, as well as 25 top-scoring sentences and abstracts (helpful in grasping the node's biological meaning). For example, the keywords for topic 0/0/0/0/0 (Table 2) strongly suggest proteolytic activities, which would be expected for the genes from the ubiquitin-mediated proteolytic pathway. The majority of the topic's MEDLINE records (102 of 119) came from the pathway's member genes; and the genes contributed most of their MEDLINE records (102 out of 138) solely to the topic. Other genes' abstracts were more broadly distributed among the topics. For example, some genes from the respiratory chain pathway contributed their abstracts mainly to two or three closely related topics (e.g. 1/0/0/0/0, 1/0/0/0/1 and 1/0/0/1/0), corresponding to mitochondrial genome, mitochondrial protein biosynthesis and mitochondrial ATP biogenesis, respectively (Table 3).

The GeneSmith module clusters genes into groups based on their topic distributions. This is the second stage of the novel 2-stage clustering process of our architecture. For example, four of the genes listed in Table 3 (Q0045, Q0105, Q0250, and Q0275) were assigned to the same group because of their similar distributions in topics 1/0/0/0/0 and 1/0/0/0/1. Although Q0130 was from the same pathway, it was clustered into another group because a significant portion of its abstracts contributed to topic 1/0/0/1/0. Even though this topic is related to the above two topics, the clustering algorithm (*k*-means) did not use this fact. The algorithm also seemed sensitive to details.

ID	Pathway	# of genes
1	Sulfur amino acid biosynthesis	14
2	Biosynthesis of sphingolipids	15
3	Respiratory chain	40
4	Pyrimidine metabolic pathway	8
5	Krebs tricarboxylic acid cycle	15
6	Cell cycle control of DNA replication	24
7	Pre-rRNA processing pathway	24
8	Ubiquitin-mediated proteolytic pathway	7
9	Early steps of protein translocation into the endoplasmatic reticulum	5
10	Vesicular protein transport in exo- and endocytosis	7
	Total	**159**
	Unique total	**155**

Table 1: Hand-picked genes from the Comprehensive Yeast Genome.

Representative keywords	Contributing genes *		Contributing groups	
proteasom, sug, rpt, atpas, **ubiquitin**, rpn, transcript, **protein**, gal, mts, tbp, **proteas**, cim, **proteolysi**, ufd, receptor, **proteolyt**, yta, tfiia, pa, channel, ms, regulatori_complex, cad, msug, conjug, famili, activ_domain, phosphoryl, transcript_activ, toa, cap, protein_degrad, nucleu, **ubiquitin_protein**, manduca, pre, nob1p, ubr, ism, fza, transcript_factor, bind_protein, muscl, gankyrin, die, nuclear, put_atpas, rna_polymeras_ii, hormone	**Gene (pathway)**	**Citations**	**Group**	**Citations**
	YGL048c (8)	38/50 (76%)	8	53/218
	YOR259c (8)	19/27 (70%)	11	46/890
	YKL145w (8)	19/22 (86%)	7	6/297
	YDL007w (8)	15/18 (83%)	9	4/287
	YDR394w (8)	6/14 (43%)	4	4/265
	YOR117w (8)	5/5 (100%)	2	3/457
	YKL216w (4)	2/29 (7%)	5	1/100
	YPR119w (6)	2/50 (4%)	0	1/199
	YLR079w (6)	2/50 (4%)	10	1/94

*Genes contributing only 1 citation to the topic were omitted.

Table 2: Representative keywords, and contributing genes and groups, for topic 0/0/0/0/0.

Topic*	Representative keywords	Description
/1/0/0/0/0 Q0045: 28/50 Q0105: 16/50 Q0130: 16/50 Q0250: 14/50 Q0275: 12/50	cystein, trna, cystathionin, petit, cys, gene, sulfur, oah, methionin, enzym, **mitochondri_genom**, **genom**, lyas, plant, **mitochondri_dna**, **clone**, serin, shlase, str, atp, sulfhydrylas, ori, homocystein, **exon**, schizosaccharomyc_pomb, acetylserin, rho, acetylhomoserin, cyp83b, **mitochondri**, nidulan, cyp83a, cytoplasm, transposit, acetyltransferas, fungi, biosynthesi, pomb, met, aspergillu, ibs, beta_synthas, **homolog**, male, glucosinol, glu, sulphur, satas, synthas, sulphat	Mitochondrial genome
/1/0/0/0/1 Q0045: 19/50 Q0105: 30/50 Q0130: 16/50 Q0250: 28/50 Q0275: 34/50	**mrna**, pet, cox, **translat**, **transcript**, gene, mutat, cob, **mitochondri**, cbp, fumaras, cox2p, synthesi, rna, oxi, nuclear_gene, nuclear, mss, ai, cox1p, aep, box, respiratori, **translat_activ**, codon, ts, nam, cbs, fum, cytochrom, suppressor, bi, arg8m, utl, excis, **mitochondri_gene**, **mitochondri_mrna**, mss51p, **protein**, phenotyp, synthet, **mitochondri_translat**, **translat_product**, cox3p, pet111p, oxa, **mitochondri_transcript**, coxiii, **pre_mrna**, suv	Mitochondrial protein synthesis
/1/0/0/1/0 Q0045: 1/50 Q0105: 0/50 Q0130: 14/50 Q0250: 4/50 Q0270: 1/50	**atp**, oscp, **atpas**, oligomycin, beta_subunit, **atp_synthas**, cox5b, cox5a, **oxygen**, sector, residu, phosphoryl, **oxid**, coq, heme, adp, hap, cyc, amino_acid, cyt, membran, oli, **mtatpas**, imp, aerob, **atpas_subunit**, f1f, **proton**, **mitochondri_atpas**, enzym, **atp_synthas_complex**, amino_acid_substitut, **atpas_activ**, vb, **yeast_atp_synthas**, som, hydrophob, acid, uas, chromatographi, aminolevulin, lethal, bovin, viia, alpha_subunit, mgi, hem, qh, **oxidas_subunit**, **mitochondri_atp_synthas**	Mitochondrial ATP biogenesis

*Other topics with 1 or 2 abstracts were omitted.

Table 3: Topic distributions of some genes from the respiratory chain pathway.

For example, while a human expert would probably group the six major contributing genes of topic 0/0/0/0/0 into a single cluster, the algorithm assigned them into three clusters (groups 7, 8 and 11 in Table 2), because their contributions ranged from 43% to 100%. Other clustering algorithms and weighting methods might improve the second clustering stage. Nevertheless, the system is robust to noise introduced throughout the process in that the results make sense despite that noise.

Discussion and Conclusion

Two-stage vs. one-stage clustering

The two-stage clustering approach, in which first MEDLINE abstracts are clustered, and those clusters are then used as input to the gene clustering stage, is a distinguishing feature of the architecture and its example implementation, GeneNarrator. The two stage design overcomes three significant drawbacks of basic single-stage clustering:

- the dominance of well-studied genes over less-studied genes,
- the dilemma of assigning less-studied genes to the right clusters, and
- the difficulty of grasping clusters' biological meanings.

Some genes are well studied, with hundreds, even thousands of hits in MEDLINE. Newly discovered and less popular genes may have only a few hits. In a basic one-stage clustering design, each gene is represented by the set of its abstracts. When directly compared, genes with many abstracts will have a strong tendency to dominate genes with only a few. This was vividly illustrated for example by (Chaussabel and Sher, 2002) (figure 2). This is a problem that needs to be addressed. The two stage design of GeneNarrator avoids this dominance problem because the number of abstracts for a gene does not translate into a corresponding degree of influence in the ultimate clustering of genes. Consequently all abstracts count, regardless of whether they discuss a gene with many other abstracts or one with only a few. This is an advantage of the two stage design as compared to a basic one stage design. Alternatively, a one-stage approach might be adopted which does some form of averaging so that gene representations do not expand without restriction as they become more studied. For example, the semantic attribute approach of Chagoyen et al., (2006) is in this paradigm.

Many genes, especially well-studied ones, are on record as participating in multiple pathways. Texts mentioning such genes may therefore contain keywords indicative of any of these pathways. In a one-stage clustering design an individual topic cluster dominated by well-studied genes may therefore discuss multiple pathways. Furthermore, sets of pathways discussed in different topic clusters may overlap. This overlap can make assigning a less-

studied gene or other gene discussed in the context of a single pathway to a cluster a dilemma. On the one hand, it must be assigned to at least one of the overlapping clusters discussing its pathway. On the other hand, it is problematic to assign it to any of those clusters, because membership in a multi-pathway cluster seems to suggest potential relevance to more than one pathway. In the two-stage approach, this problem is less likely. During the first stage, text clustering may group abstracts into topics containing more than one pathway, and one pathway may end up in more than one topic. But the second, gene clustering stage permits assigning less-studied genes to individual pathways, while well-studied genes can be assigned to multiple topics simultaneously.

Finally, each topic cluster should be annotated, such as with a list of representative keywords, sentences, etc. From a user's perspective, it will typically be easier to grasp the biological meaning of a gene cluster from keyword and sentence annotations associated with it, if it contains truly related genes.

Availability and Requirements

System developers may wish to incorporate two-stage clustering and other aspects of the architecture discussed in this paper into their own systems. Alternatively, the code we developed is available to use as a starting point. The official project Web site URL is http://bioinformatics.ualr.edu/dan/genenarrator/. Links are provided for downloading source code, a zip package, a sample data set, and the user manual. Linux is needed as CrossBow requires it, however, other components can run under other platforms. One GB is required for moderately-sized data sets (~500 genes/proteins/metabolites or ~10,000 MEDLINE abstracts).

Acknowledgements

This work was supported in part by The Procter and Gamble Company. We are grateful for the comments of reviewer #1.

References

1. Adryan B, Schuh R (2004) Gene Ontology-based clustering of gene expression data. Bioinformatics 20: 2851-2852. » CrossRef » Pubmed » Google Scholar

2. Badea L (2003) Functional discrimination of gene expression patterns in terms of the gene ontology. In Pac Symp Biocomput 8: 565-576. » CrossRef » Pubmed » Google Scholar

3. Becker K, Hosack D, Dennis G, Lempicki R, Bright T, et al. (2003) PubMatrix: a tool for multiplex literature mining. BMC Bioinformatics 4: 61. » CrossRef » Pubmed » Google Scholar

4. Chagoyen M, Carmona-Saez P, Shatkay H, Carazo JM, Pascual-Montano A (2006) Discovering semantic features in the literature: a foundation for building functional associations. BMC Bioinformatics 7: 41. » CrossRef » Pubmed » Google Scholar

5. Chaussabel D, Sher A (2002) Mining microarray expression data by literature profiling. Genome Biology 3: 0055.1-0055.16. » CrossRef » Pubmed » Google Scholar

6. Ding J, Berleant D (2005) MedKit: a helper toolkit for automatic mining of MEDLINE/PubMed citations. Bioinformatics 21: 694-695. » CrossRef » Pubmed » Google Scholar

7. Ding J, Hughes LM, Berleant D, Fulmer AW, Wurtele ES (2006) PubMed Assistant: a biologist-friendly interface for enhanced PubMed search. Bioinformatics 22: 378-380. » CrossRef » Pubmed » Google Scholar

8. Entrez Programming Utilities. http://eutils.ncbi.nlm.nih.gov/entrez/query/static/eutils_help.html.

9. Frank E, Hall M, Trigg L, Holmes G, Witten IH (2004) Data mining in bioinformatics using Weka. Bioinformatics 20: 2479-2481. » CrossRef » Pubmed » Google Scholar

10. Glenisson P, Antal P, Mathys J, Moreau Y, De Moor B (2003) Evaluation of the vector space representation in text-based gene clustering. Pacific Symposium on Biocomputing 8: 391-402. » CrossRef » Pubmed » Google Scholar

11. Hofmann T (1999) The Cluster-Abstraction Model: unsupervised learning of topic hierarchies from text data. Proceedings of the International Joint Conference on Artificial Intelligence. » CrossRef » Google Scholar

12. Homayouni R, Heinrich K, Wei L, Berry MW (2005) Gene clustering by latent semantic indexing of MEDLINE abstracts. Bioinformatics 21: 104-115. » CrossRef » Pubmed » Google Scholar

13. Joslyn CA, Mniszewski SM, Fulmer A, Heaton G (2004) The Gene Ontology Categorizer. Bioinformatics 20: i169-177. » CrossRef » Pubmed » Google Scholar

14. Kankar P, Adak S, Sarkar A, Murari K, Sharma G (2002) MedMeSH Summarizer: text mining for gene clusters. Proceedings of the Second SIAM International Conference on Data Mining. » Google Scholar

15. Kennedy PJ, Simoff SJ, Skillicorn D, Catchpoole D

(2004) Extracting and explaining biological knowledge in microarray data. The 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining. »CrossRef » Pubmed » Google Scholar

16. Khatri P, Draghici S (2005) Ontological analysis of gene expression data: current tools, limitations, and open problems. Bioinformatics 21: 3587-3595. »CrossRef » Pubmed » Google Scholar
17. Kim CC, Falkow S (2003) Significance analysis of lexical bias in microarray data. BMC Bioinformatics 4: 12.»CrossRef » Pubmed » Google Scholar
18. Manning CD, Schutze H (1999) *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts. »CrossRef » Google Scholar
19. Mao X, Cai T, Olyarchuk JG, Wei L (2005) Automated genome annotation and pathway identification using the KEGG Orthology (KO) as a controlled vocabulary. Bioinformatics 21: 3787-3793. »CrossRef » Pubmed » Google Scholar
20. Masys DR, Welsh JB, Fink JL, Gribskov M, Klacansky I, et al. (2001) Use of keyword hierarchies to interpret gene expression patterns. Bioinformatics 17: 319-326. »CrossRef » Pubmed » Google Scholar
21. McCallum AK (1996) Bow: a toolkit for statistical language modeling, text retrieval, classification and clustering. http://www-2.cs.cmu.edu/~mccallum/bow/.
22. Munich Information Center for Protein Sequences, Comprehensive Yeast Genome Database. http://mips.gsf.de/proj/yeast/pathways/.
23. Oliveros JC, Blaschke C, Herrero J, Dopazo J, Valencia A (2000) Expression profiles and biological function. Genome Informatics 11: 106-117. »CrossRef » Pubmed » Google Scholar
24. Pasquier C, Girardot F, de Fombelle KJ, Christen R (2004) THEA: ontology-driven analysis of microarray data. Bioinformatics 20: 2636-2643. »CrossRef » Pubmed » Google Scholar
25. Porter MF (1980) An algorithm for suffix stripping. Program 14: 130-137.»CrossRef » Google Scholar
26. Raychaudhuri S, Altman RB (2003) A literature-based method for assessing the functional coherence of a gene group. Bioinformatics 19: 396-401. »CrossRef » Pubmed » Google Scholar
27. Raychaudhuri S, Schutze H, Altman RB (2002) Using text analysis to identify functionally coherent gene groups. Genome Res 12: 1582-1590. »CrossRef » Pubmed » Google Scholar
28. Renner A, Aszodi A (2000) High-throughput functional annotation of novel gene products using document clustering. In Pac Symp Biocomput 5: 54-68. » Pubmed » Google Scholar
29. Robinson PN, Wollstein A, Bohme U, Beattie B (2004) Ontologizing gene-expression microarray data: characterizing clusters with Gene Ontology. Bioinformatics 20: 979-981.»CrossRef » Pubmed » Google Scholar
30. Rose K, Gurewitz E, Fox G (1990) Statistical mechanics and phase transitions in clustering. Phys Rev Lett 65: 945-948.»CrossRef » Pubmed » Google Scholar
31. Shatkay H, Edwards S, Wilbur W, Boguski M (2000) Genes, themes, and microarrays: using information retrieval for large-scale gene analysis. In 8th International Conference on Intelligent Systems for Molecular Biology (ISMB) 8: 317-28.»CrossRef » Pubmed » Google Scholar
32. Slaton G, McGill MJ (1983) *Introduction to modern information retrieval*. McGraw-Hill, New York, NY.
33. Smid M, Dorssers LCJ (2004) GO-Mapper: functional analysis of gene expression data using the expression level as a score to evaluate Gene Ontology terms. Bioinformatics 20: 2618-2625. »CrossRef » Pubmed » Google Scholar
34. Tanabe L, Scherf U, Smith LH, Lee JK, Hunter L, et al. (1999) MedMiner: an internet text-mining tool for biomedical information, with application to gene expression profiling. BioTechniques 27: 1210-1217. »CrossRef » Pubmed » Google Scholar