This is to certify that the
thesis entitled

MINING INTERESTING RULES BY COMBINING KNOWLEDGE DISCOVERY FROM DATA WITH LITERATURE BASED DISCOVERY

presented by

Samah Jamal Fodeh

has been accepted towards fulfillment
of the requirements for the

Master of Science degree in Computer Science and Engineering

Major Professor's Signature

8/3/2006

Date

MINING INTERESTING RULES BY COMBINING KNOWLEDGE DISCOVERY FROM DATA WITH LITERATURE BASED DISCOVERY

By

Samah Jamal Fodeh

A THESIS

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

MASTER OF SCIENCE

Department of Computer Science and Engineering

2006

ABSTRACT

MINING INTERESTING RULES BY COMBINING KNOWLEDGE DISCOVERY FROM DATA WITH LITERATURE BASED DISCOVERY

By

Samah Fodeh

The application of data mining and knowledge discovery techniques to real world domains is a rewarding but extremely challenging process. The massive size, high dimensionality, and fast growing databases are among the key challenges that need to be addressed by these techniques. In this thesis, we investigate the problem of mining interesting association rules from large databases in the presence of background knowledge. We hypothesized that the background knowledge provides useful domain information that may help us to subjectively evaluate and interpret the extracted rules. To illustrate the advantages of incorporating background knowledge, we developed a framework called *MIR* for mining interesting rules from the medical domain. We showed that the background knowledge helps us to distinguish obvious rules from those that are either spurious or probable. We tested the MIR framework using deidentified Electronic Medical Records (EMR) data from MQIC (Medical Quality Improvement Consortium) and utilized background knowledge available in the PubMed bibliography database

Acknowledgments

I would like to thank Dr. Pang-Ning Tan, my major advisor, for his help and thoughtful advice over my thesis work. Also, I would like to thank my committee members Dr. Abdol-Hossein Esfahanian and Dr. Joyce Chai for agreeing to serve on my committee. I also would like to thank Dr. Henry Barry, Dr. Michael Zaroukian, and Dr. David Weismantel from College of Human Medicine for providing us with the data and the domain expertise needed to analyze the results of our experiments.

Also, I wish to express my appreciation to my parents and husband for their support and love which gave me the inspiration and persistence to overcome all the difficulties I had over all the times of working on my thesis.

Table of Contents

Chapter 1

Introduction

As we are transitioning from an era of information dearth to information glut [36], there is a need to explore the massively growing data in order to extract precious pieces of information. Such exploration process is referred to as knowledge discovery from data (KDD). Knowledge discovery is the intelligent science of mining and extracting previously unidentified but potentially valuable information embedded in the data [32]. It is also the process of searching large databases for *models* and *patterns* that describe the structure of the data by applying techniques from statistics, machine learning, and pattern recognition. According to Hand [12] a model is a global representation of a structure that summarizes the systematic component underlying the data or that describes how the data may have arisen. The word "global" here signifies that it is a comprehensive structure referring to many cases. In contrast a pattern is a local structure relating to just a handful of variable and a few cases. Examples of data mining patterns include outliers, clusters, frequent itemsets and association rules, whereas classification and regression models are examples of the discovered models.

Over the past two decades, knowledge discovery has emerged as a key enabling technology for a variety of applications, from business intelligent and threat analysis to medical informatics and the modeling of Earth Science data. Despite its remarkable success, existing data mining technology is often plagued by several critical and fundamental challenges:

- The number of extracted patterns tends to overwhelm our ability to analyze them. For example, there has been tremendous interest in integrating a data mining technique known as anomaly detection into intrusion detection systems. However, the number of alarms generated by such systems may exceed the ability of security experts to administer them.
- Lack of information to interpret the significance of data mining results. For example, in association analysis, which is a widely used technique for finding strongly correlated features in data, some patterns may seem novel to domain experts while others may be obvious or even spurious. Though statistical methods such as the chi-square test have been successfully applied to eliminate spurious patterns, separating the novel from obvious patterns remains a significant barrier for many data mining systems.

A common theme among those challenges is the need to incorporate background knowledge directly into the data mining process. Background knowledge may help domain experts to order the extracted patterns so that it is sufficient for experts to examine only the highest ranked patterns. In addition, it may also provide the information needed to aid domain experts in interpreting the meaning and significance of the patterns. While there has been considerable efforts to incorporate background knowledge into various learning tasks, such knowledge must first be hand coded by domain experts.

Recently, there has been considerable interest in developing text mining techniques for extracting knowledge directly from literature, such as the digital libraries and other repositories of scholarly articles. Such a process is known as literature-based discovery [42][43][16][17][23][33][38]. The goal of literature-based discovery is to

uncover potentially meaningful relations between various concepts in a given application domain using information from the literature. A classic example of literature-based discovery is given in [34], by Swanson, who had successfully discovered an implicit relation between fish oil and Raynaud's disease by manually searching journal articles in the medical domain.

The objective of this thesis is to incorporate background knowledge automatically into the knowledge discovery process. Combining knowledge discovery from data with background knowledge is a challenging task due to several reasons. First, each knowledge type has its own representation. For instance, the knowledge discovered from data may be encoded in the form of association rules whereas the background knowledge may be contained in the vast amount of scientific literature. The differences in their representations complicate the integration task because each rule extracted from the data must be mapped to its corresponding concepts in the literature. Second, the background knowledge may be unstructured and high in volume, thus requiring extensive preprocessing steps. Third, a systematic approach is needed to ensure that incorporating background knowledge actually helps to improve the quality of models and patterns discovered from data.

This thesis focuses on utilizing background knowledge (in the form of structured ontologies or scholarly articles in the scientific literature) to evaluate and interpret the association rules extracted from large databases. To illustrate the advantages of using background knowledge, we developed a framework called *MIR* for mining interesting rules from the medical domain. We showed that the background knowledge helps us to distinguish obvious rules from those that are either spurious or probable. The probable

rules correspond to unexpected relationships between two concepts that are indirectly associated with each other (see Chapter 4). We tested the MIR framework using deidentified Electronic Medical Records (EMR) data from MQIC (Medical Quality Improvement Consortium) and utilized background knowledge available in the PubMed bibliography database.

The remainder of this thesis is organized as follows. In Chapter 2, we describe the various representations of background knowledge and introduce the literature-based discovery problem. This chapter also presents other existing work on utilizing background knowledge for knowledge discovery. In Chapter 3, we provide a brief overview on the association rule mining problem, while Chapter 4 summarizes the key components of our proposed MIR framework. In Chapter 5, we demonstrate the results of experiments that were performed to evaluate the effectiveness of the MIR framework. Finally, Chapter 6 gives the conclusions and suggestions for future work.

Chapter 2

Literature-Based Knowledge Discovery

The literature is a rich and major source of information in different areas of science. It is defined in Webster's dictionary as "The collective body of literary productions, embracing the entire results of knowledge and fancy preserved in writing; also, the whole body of literary productions or writings upon a given subject, or in reference to a particular science or branch of knowledge, or of a given country or period." Another definition of literature that is closer to our work is given in answers.com, which describes the literature as the body of written work produced by scholars or researchers in a given field. In this thesis, we will use online literature as the background knowledge that will be incorporated into the process of data mining. Online literature is the digital version of the written body of literature posted on the web to be available for the public use and to serve in different purposes such as the automated knowledge discovery systems. Background knowledge extracted from online literature may help domain experts to evaluate the extracted patterns from the data, and it may also provide the information needed in interpreting the meaning and significance of the patterns.

First, we describe the various ways to represent background knowledge. We then describe literature-based discovery which is the process of searching for hidden and important connections among information embedded in published literature [9]. Finally,

we provide a survey on the prior work on how to incorporate background knowledge into data mining.

2.1 Representation of Background Knowledge

With the accelerated growth of the Internet, knowledge from diverse domains has been stored and made accessible for online access. Since acquiring information from the Web tends to be cheaper than acquiring it from human experts, considerable interests have emerged to employ this information as background knowledge for knowledge discovery systems. Background knowledge is generally represented in two different forms: *structures* and *unstructured* background knowledge. We will discuss these representations in the remainder of this section.

2.1.1 Structured Background Knowledge

Structured background knowledge means that the relevant pieces of information are encapsulated into different blocks that are semantically related to form tabular or a hierarchical structure such as ontologies. The advantages of using structured background knowledge are: (1) it is reliable because the structure was built upon the agreement of domain experts, which made it a strong reference for research purposes, (2) the simplicity of retrieving the required information from the hierarchy, and mapping the to-be-classified data to the hierarchy due to the semantically related features between its blocks. These advantages made researchers favor using the structured model over the unstructured one. However, structured background knowledge has a drawback since it

remains limited in scope and static in its current contents for a considerable duration of time because it is not frequently updated.

Ontology is an example of structured background knowledge. The objective of an ontology is to define the concepts and relationships in the domain. Such a conceptualization is intended to help with automatic translation among existing databases, because it can be used to form a bridge that associates different databases together. In the rest of this section we will provide some examples of structured background knowledge.

Unified Medical Language System (UMLS):

UMLS is a medical comprehensive compendium that contains many vocabularies from different sources and languages [41], and provides a mapping structure between those vocabularies. UMLS consists of three major sections: The first section is called Metathesaurus which contains different concepts (a concept may represent different synonyms of the term from different languages). A group of related concepts are mapped to a semantic type. The second section is the semantic network. There we can find the semantic relations between different semantic types. Finally we have the specialist lexicon, which is developed to provide the lexical information needed for Natural Language Processing (NLP) specialist, such as the syntactic, morphological, orthographic information of the word.

Gene Ontology:

Gene Ontology: (GO) is one of the most popular ontologies for bioinformatics research. It is a controlled vocabulary of terms in cellular and molecular biology; it consists of three sub ontologies - the molecular functions of gene products, the biological process,

and the localization of the components. This ontology is used in the process of analyzing the results of the micro array experiments due to the huge results that yield from those experiments, which need to be organized and classified in a way that makes it easier for the researcher to interpret and read these results.

WordNet:

Wordnet is a lexical reference system or linguistic dictionary that contains English nouns, verbs, adjectives, and adverbs which are organized into synonym sets, each representing a lexical concept, and records the semantic relations that links the synonym sets together. WordNet can be explored in different directions, finding words that mean the same, words that are more general, more specific, as well as getting a brief description. This semantic lexicon can be used as a dictionary as well as supporting automatic text analysis.

Semantic Web:

Semantic web is another way of representing knowledge; it aims to make the tremendous information in literature machine processable on a large scale [25]. This requires the data to be standardized format wise, to make it easily interchangeable. So the goal of the semantic web is to enhance the usability of the web and its resources by marking the different documents with semantic information [41]. This enables computers to read and associate different documents together. Ontology's are one example of the semantic Web vision because they encode data into sets of thesauri and records the semantic relations between these sets.

2.1.2 Unstructured Background Knowledge

The unstructured background knowledge could be in the form of video, images, or large body of texts. Unlike structured knowledge, having the information encoded in a text format made it wealthier due to the very frequent update of the contents in the unstructured representation of background knowledge. Nevertheless, processing the text is more challenging because it requires special Natural Language Processing algorithms for this purpose, whereas the organized form of the structured representation made retrieving and manipulating the information easier. Examples of unstructured background knowledge include Pubmed, which is one of the most popular sources of texts in the biomedical domain, and Wikipedia, which is also a huge general encyclopedia that contains definitions in a free text format.

2.2 Literature-Based Knowledge Discovery

Literature-Based discovery (LBD) is the process of searching for hidden and important connections among information embedded in published literature. The linkages and associations are found by drawing paths between two different objects. Most of the works in LBD follow two approaches for knowledge discovery—open or closed discovery [39].

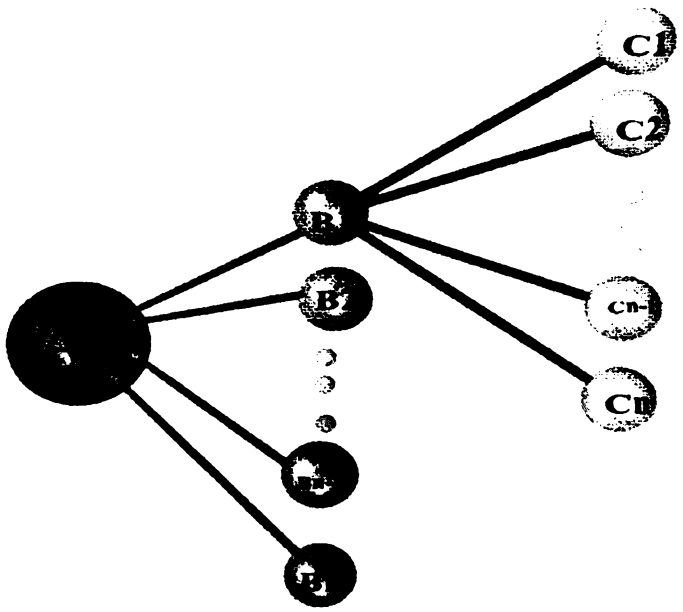

Figure 1: Open discovery approach.

Open discovery is the task of generating a hypothesis based on a given starting concept [39]. The exploration process attempts to find an association between a starting concept (A) and the "to be discovered" concept (C). The starting concept A is used as a query to an online database and all the documents that include this concept are retrieved. The most important terms and phrases are then extracted using certain text processing technique. The list of these important terms is then filtered to get an intermediate list of B-concepts. The intermediate concepts represent topics that are closely related to the starting concept A. The literature is then queried again using each of the B concepts. The resulting documents are processed to get the most important terms. These terms correspond to candidates for the target concept C. The list of candidates for target concept C is then filtered via manual inspection by the domain experts. As a result of this processing, a new hypothesis could be generated that the concept C is indirectly related to the concept A [9].

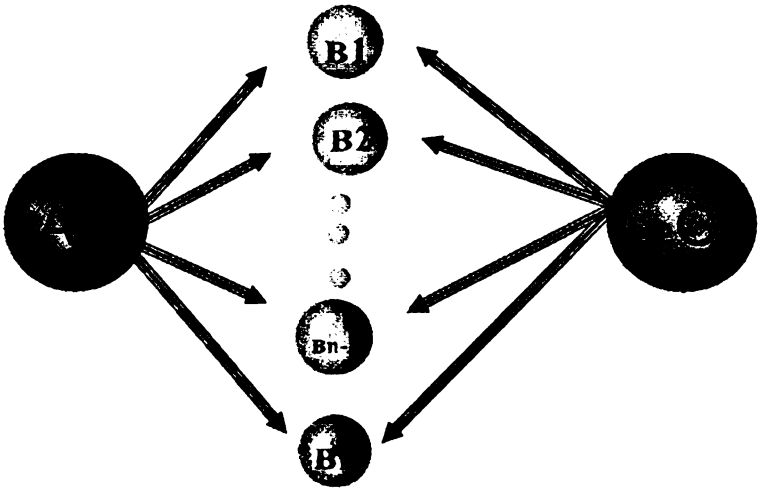

Figure 2: Closed Discovery approach

Unlike open discovery, the closed discovery approach starts with known A and C concepts. The relation between the concepts A and C could be a previously generated hypothesis or an observed association. The goal of the closed discovery approach is to find novel B concepts that explain the initial observation or hypothesis [9].

2.2.1 Discovery of Associations in Literature

As explained in Section 2.2, the goal of using literature-based discovery is to find an association between terms A and C via an intermediate term B. The latter relation is quite challenging to discover, because of the difficulty in selecting the intermediate concepts B_i from the list of discovered B concepts. Therefore, we need a measure to quantify and rank the relations (A, B_1), (A, B_2).....(A, B_n) and the relations (B_1,C),(B_2,C)...(B_n,C) and choose the highest ranked concept B_i. Structured ontology and semantic relations may be used to quantify the semantic relations (A, B_i) and (B_i,C).

There are several studies that have recently been conducted for finding associations in the literature. For instance, the work in [42] looks for implicit indirect associations between two concepts in the medical domain using Medline records. To evaluate the candidate relations, the authors have used the mutual information measure to determine the strength of associations between (A,B) and (B,C). Another approach is suggested in [43], where the authors seek to find implicit relations between two medical concepts (A,C) via an intermediate concept B. First, they compute the co-occurrences of (A,B) and (B,C) using Pubmed abstracts. They then evaluate the semantic relations between these concepts using the UMLS ontology. In the following subsections, we will give an overview of these related works.

Extending the Mutual Information Measure to Rank Inferred Literature Relationships. [42]

In this work, medical literature is processed to build a network of associations between two selected objects, such that no direct association exists between these objects and an implicit kind of association is to be revealed.

Identifying Literature-Based Associations:

The common approach used to associate terms in a text, is searching for their co-occurrences, but this technique has some drawbacks due to the fact that those linkages may be very general and useless, since no valuable information is translated. Additionally, the resulting relationships may be false alarms, because there is no apparent association between the terms [42].

Mutual Information Measure (MIM):

This measure was proposed by [42]. It measures the strength of associations between terms in literature and is widely used to quantify dependencies between variables including co-occurring terms in text.

$$MIM(A,B) = \left(\frac{P_{AB}}{P_A . P_B} \right)$$

Where P(AB) is the probability of A ,B appearance in the same document, and P(A),P(B) are the probabilities of A ,B appearance in the document, respectively.

Scoring the inferred associations:

The author in this paper proposes to extend the MIM measure to find sub relations between A→B and B→C by computing their MIM score and then combine the results following tow different methods:

- Averaging the MIM (AMIM): scores for A→ B_n and B_n →C, then divide the result by the total number of connections between A and C to normalize.

$$Score(A,C) = \sum_{n=1}^{t} \left(\frac{\left\{ \left(\frac{P_{ABn}}{P_A . P_{Bn}} \right) + \left(\frac{P_{BnC}}{P_C . P_{Bn}} \right) \right\}}{2t} \right)$$

- Minimum value of MIM (MMIM): computes the minimum value within the (t) connections between A,C.

$$Score(A,C) = \frac{\sum_{n=1}^{t} \min \left\{ \left(\frac{P_{ABn}}{P_A . P_{Bn}} \right) + \left(\frac{P_{BnC}}{P_C . P_{Bn}} \right) \right\}}{t}$$

As mentioned by the author the results form MMIM is slightly better than the AMIM.

Experiments were done using MEDLINE which is an immense online medical database, created by the National Library of Medicine. The term capsaicin was examined and their tests revealed associations between capsaicin and other terms like calcium, neurons, and ileums. Those associative patterns were ranked by the proposed method MMIM and the shared relations (where capsaicin and other terms share the same documents). When using a shared relationship, the general relationships tended to be on the top of the list, while using MMIM score placed the famous and obvious relations on the top of the list. Many of the research methods in data mining depend on querying and screening available information. While this method is searching for associations between two different terms, which is a small network of one level, it was mentioned that this could be extended to more levels, but did not discuss how that could be done. Because there are some challenges in the next levels (whether to keep (A, C) as the references while computing MMIM in the next levels or choose new terms from the top n-list).

A Semantic Approach for Mining Hidden Links from Complementary and Non-interactive Biomedical Literature. [43][16][17]

In this method, the authors hypothesize that considering complementary and non-interactive biomedical literature together can disclose valuable knowledge that is not apparent if considering each individually. Bio-sbKDS is a semantic based algorithm that discovers hidden relations between biomedical concepts; it uses (UMLS) ontology for this purpose. Their method finds the relation C→A, by finding the intermediate relations C→B and B→A.

The algorithm requires the user to input the kind of relationship he is aiming to, (Initial Semantic Relation (ISR)) between the starting concept C and the to-be-discovered

concept A. They continue with their algorithm by finding the semantic type of C (ST_C), and then they find all the semantic types that are related to C through different relations in UMLS and call it ST_B_can. On the other hand, they use the relation ISR to identify the semantic type of the to-be-discovered concept A (ST_A), then extend (ST_A_can) semantic set through the is-a relations in UMLS and call the resulting set (ST_A_can_ext). After that, they do the filtering by examining if there are semantic relations between the semantic types (ST_B_can) and (ST_A_can_ext). This step is performed using semantic relations from UMLS, the resulting sets will be (ST_A) and (ST_B). At this point, they incorporate literature again, using text-mining this time, by searching Pubmed abstracts against the concept C, to get the B_can list from the MeSH Headings of the abstracts. Now they apply the (ST_B) upon the B_can list, to exclude irrelevant terms that do not belong to any semantic type in (ST_B) set, the terms are ranked and the top n are selected. The same procedure is applied to find the A candidates but using the top B terms from the previous step as the query search terms against the abstracts. For the final step, they extract the A candidates terms that co-occur with C in Pubmed abstracts. Their experiments showed that their algorithm is more efficient. It was compared with LSI-Based (Latent-Semantic Indexing) which uses singular-value decomposition analysis and cosine measure between C,B and B,A for ranking and AR-Based (Association Rule Based) which generates all C→B,B→A rules, and applies the transitive law to discover the hidden link.

As described above, the method depends on UMLS. Because it is a famous ontology that contains reliable medical information, it will discover the association between A→C only if the relations between A→B and B→C are available in UMLS.

Actually this step could be considered an advantage because it makes the results more trustworthy since it prunes away the irrelevant terms, however, there is a negative side about it, because it bounds the performance of the method to the UMLS knowledge which is a learning ontology updated constantly. Another drawback is related to the cost of this method, due to the frequent querying of Pubmed literature.

Supporting Discovery in Medicine by Association Rule Mining in Medline and UMLS [14]

Literature is incorporated in this work to evaluate association rules using a new system which is the Discovery Support System. This is an interactive system that demonstrates how an embedded relation could be found between a starting concept and the concepts of interest. More specifically, finding relations between X→Z and Z→Y reveals an embedded relation between X→Y that is not available in literature. Medline database is used along with MeSH descriptors which are terms assigned to the Medline records as a representation of contents of the articles [14].

The first step is to prepare the association rules between the MeSH concepts. The next step is to find all the concepts *Y* related to the starting concept *X* (*X* could be any concept, i.e. a disease, and *Y* could be a symptom), then all the concepts *Z* related to *Y* are found [14]. *Z* could be a chemical substance that alleviates the symptoms *Y*. In the last step, they check X→Z co-occur in literature, if this association X→Z does not appear in literature, then this is a new discovered relation that is determined by a human intervention or a clinical investigation [14]. Additionally, they add a limiting or filtering procedure on the relations between X→Z. This could be implemented by several

methods. One way is to incorporate the semantic types from UMLS or to add a threshold on the support and confidence of the associated rules X→Y, Y→Z.

In their experiment, they divided the Medline database and the corresponding association rules into tow segments according to the publication dates of the documents [14]. The older segment is within the period (1990-1995) and the new segment is within the period (1996-1999). This is because they wanted to evaluate their results by using the older segment to mine for the expected associations as new discoveries and then examine the existence of these associations in the new segment articles. One advantage of this method is the reliability because of imposing semantic relations from UMLS to qualify the predicted relation X→Z. On the other hand, using thresholds for support and confidence for the intermediate relations X→Y and Y→Z restricts the results to the value of the selected thresholds.

Improving Literature Based Discovery Support by Background Knowledge Integration [15]

This work is an extension of the previous work, but the authors here incorporate different databases from literature to reach their goal. In this work, they define UMLS as the main source for the relations between the biomedical concepts. Those relations are extracted and stored in a knowledge base. In this work they try to find candidate genes of a certain disease. To achieve this, they include different sources of knowledge.

- The chromosomal locations for the diseases are available in OMIM (Online Mendelian Inheritance in Man;) a database catalog of human genes and genetic disorders and contains textual information and references.

- The gene locations are extracted from HUGO(a database that contains genes names and symbols), LocusLink, and OMIM.
- The disease location of manifestation (the tissue or organ where the disease phenotype is expressed) from OMIM and Medline. [15]
- The gene expression location from the Uni Gene database.

They use background knowledge to connect the dots between the disease of interest and its causing-genes. This resembles the goal of [37] but in this work they don't include an anatomical ontology, instead they use a lower more specific level of chromosomal ontologies. They have a disease as a starting concept and mine for associations in Medline guided by semantic types and relations. Apparently, the associations should tie a disease with chromosomal and manifestation locations as well as linking the candidate genes with their chromosomal and expression location. If the regions or locations match in any way, this could be a good indication that this is a disease causing gene.

Other different works used literature for deriving association rules such as the work in [32] [33]. In this work by Swanson and others; They looked for implicit associations between terms (A,C) by finding associations between the concepts (A,B) and (B,C), where a direct relation between (A,C) is non-existent. They use MeSH as indexes to search Medline. They claim that the coherence of retrieved literature is necessary for the accuracy of the results. Coherence means the literature on magnesium, for example, should not include articles that mention magnesium only incidentally [32]. This literature cohesiveness is obtained using MeSH terms. Furthermore, ranking the top n-list of the intermediate B terms depends on the density of the common MeSH terms between AB

and BC records. Likewise in [33] authors discussed the prediction of relationships in scientific Literature.

2.3 Combining Background Knowledge from Literature with Data Mining

The knowledge available in literature has been incorporated in exploring different domains as we mentioned above. We can find good examples of this type of research in the medical domain. Incorporating background knowledge; that is available in the online literature; directly in the data mining process supports the performance of the automated knowledge discovery systems and positively affects their results. It will aid domain experts to order the extracted patterns in order to examine only the highest ranked patterns. In addition, it also provides the information needed to aid domain experts in interpreting the meaning and significance of the patterns. Background knowledge is utilized in analyzing association rules, clustering and classification. In the next subsections, we will demonstrate some of the examples of this work.

2.3.1 Using Background Knowledge for Association Analysis

The background knowledge that resides on the World Wide Web is used to build bridges between various categories and blocks in the targeted databases. The following articles are examples of this work.

Integration of text- and data-mining using ontology's successfully selects disease gene candidates [37]

In this paper, the authors' goal is to identify disease genes candidates, by using an anatomical ontology to integrate text-mining of biomedical literature and data-mining of gene expression data to identify candidate disease genes.

Figure 3 summarizes the proposed algorithm. In effect, they try to build a bridge between tow different databases using Evoc ontology (a controlled vocabulary which describes the human anatomy).

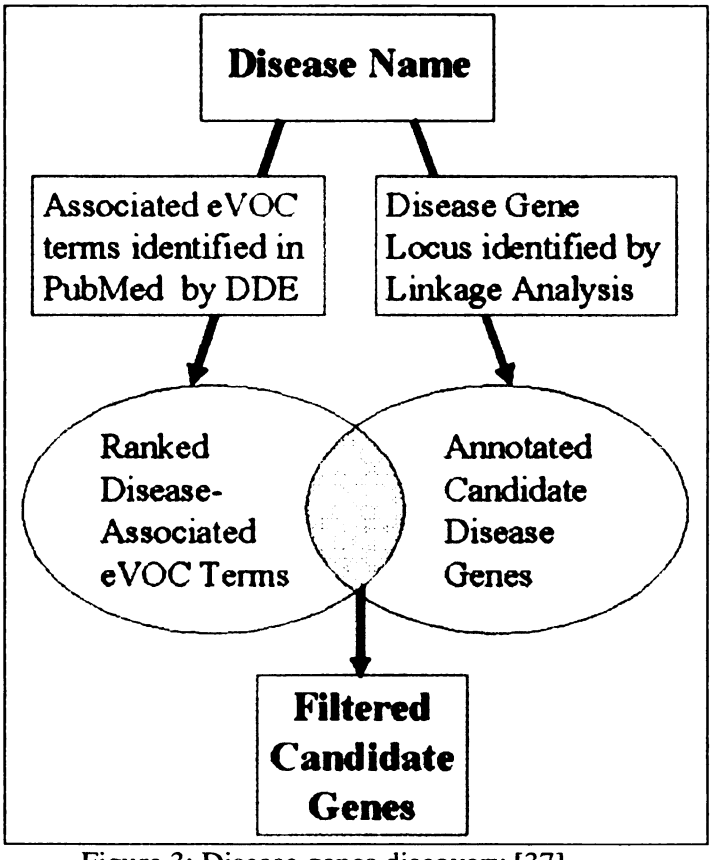

Figure 3: Disease genes discovery [37].

The database on the left is the Pubmed abstracts, while the database on the right is the disease genes from the ensemble online database. Their method starts with processing the

Pubmed abstracts in literature, and finding the Evoc terms that co-occur with the disease name. Similarly, selected genes from the ensemble database are annotated with an Evoc term from the Evoc Ontology. After that, they rank the selected anatomy terms by computing the frequency of association and the frequency of annotation for each term, the (N) top evoc terms are selected from the list and then intersected with the evoc terms annotated with the candidate disease genes selected from the ensembl database. The intersection of both sets should yield at least (N) evoc terms that match the top (N) associated terms with the disease name in the Pubmed abstracts. The ranking function, as mentioned, depends on two coefficients, the frequency of association that is the number of abstracts containing the Evoc term and the disease name divided by the total number of abstracts containing the disease name. The frequency of annotation is the sum of all annotated genes at the node and descendants of the evoc term in the evoc ontology hierarchy, divided by the total number of annotated genes.

$$\text{Ranking Score} = [2*f(\text{association}) + f(\text{annotation})]/2 .$$

It is clear that the method depends on the Evoc ontology when integrating both databases which is a reasonable step to execute, because initially ontologies are created after intensive and reliable studies that makes them robust references for researchers.

Using an Interest Ontology for Improved Support in Rule Mining. [44]

In this work, the authors use interest ontology in order to extract association rules that associate ages and gender with interests on the higher level. So, interests in the data that are specific are replaced by more general interests on the higher levels of the ontology hierarchy. The ontology used is an interest ontology that consists of interest hierarchies based on Yahoo and ICQ [44]. There are 11 levels of interests in the interest hierarchy.

The data is collected from different home pages and consists of records of real personal information that contains demographic and interest data [44]. Basically, the algorithm stores the cleaned data (collected from the web) in an object relational database. After that association rules are created using WEKA, a tool that implements different data mining techniques. A problem emerged here that most people express their interests on the lower level of the interest hierarchy, which negatively affected the support of the rule. In order to overcome this problem and to get higher accuracy and support [44], a support raising algorithm was developed which includes the records from the database that are mapped to the lower levels of the interest hierarchy.

2.3.2 Using Background Knowledge for Clustering

The impetus behind using ontologies in clustering is its organized categorized structure which encodes the similar objects under certain conditions within one block at a certain level. It breaks this block into subdivisions in the lower levels, because of adding more conditions or attributes to the objects. For example, the research in [22] puts the genes with similar expression profiles into one category because this similarity implies strong correlation in their biological process. They evaluate their clusters by mapping them to GO categories by computing the probability of the hyper geometric distribution which measure whether a cluster is enriched with genes from a particular category to a greater extent than that would be expected by chance [22]. So the computed probability would give a flavor if the genes in the cluster have the same biological function or not.

Another example of ontology driven clustering is in [40]. The proposed approach uses similarity/distance measure on the items in the GO hierarchy. The similarity

between two terms c_i, c_j in the hierarchy is computed using Lin's similarity model. Where $S(c_i, c_j)$ are the set of parent terms shared by both c_i, c_j. This algorithm was used for clustering gene records using the GO hierarchy and it gave good results. In the next subsection, we review some of the work that incorporates background knowledge in order to cluster the data.

Gene Ontology Friendly Bi-clustering of Expression Profiles [22] [21]

This work incorporates Gene Ontology (GO) from literature, to perform a biclustering approach. Before continue explaining the method, here is a brief definition of GO.

GO is a directed acyclic graph, GO=<V,E>, where V is a set of gene function description (GO terms) and E is the relationship on V [22]. A relationship between a term and its ancestor is a part-of relationship. As a result of this ontology representation, genes that are annotated with a GO term are subset of genes annotated with the term's ancestor. Formally, if T is a GO term with annotated genes O, and T' is the ancestor of T with annotated genes O', $O \subseteq O'$. The authors hypothesize that genes with similar expression profiles implies strong correlation in their biological process; where an expression profile of a gene is the process by which a gene's DNA sequence is converted into the structure and functions of a cell [41]. Moreover, they propose that clustering the genes under all conditions or expression levels will separate the biologically related genes from each other [22]. As a result, the clusters they are aiming for should have the maximal number of genes that have coherent tendency (consistent ranks) along a subset of conditions and these are called Tendency-Preserving Clusters (TP). The proposed algorithm Smart Hierarchical Tendency Preserving Clustering (SHTP-Clustering) is based on TP-clustering model that forms the Tendency Preserving tree (TP-tree) and incorporates GO

categories in the process such that each sub tree in the TP-tree has a matching sub tree in the GO hierarchy.

To define tendency between genes, a ranking function is defined. It translates the conditions in a gene's expression profile into an alphabetical ordered sequence. Genes with consistent ranks under a subset of conditions or expression levels are grouped in one TP-cluster. After that they build their objective TP- tree using the TP-clusters such that each node is a TP-cluster by itself, and the nodes at level m correspond to m dimensional clusters.

Annotation of gene cluster:

After having the TP-tree ready, the produced Gene TP-clusters are annotated using GO. To achieve this annotation or mapping, they inspect the enrichment of the functional categories (TP-clusters) by introducing a P-value of the hyper geometric distribution, the P-value measures whether the cluster is enriched with genes from a particular category to greater extent than that would be expected by chance[22].

$$P = 1 - \sum_{i=0}^{k} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}}$$

The probability of observing at least (k) genes from a cluster of (n) genes by chance in a category containing (f) genes from a total of (g) genes.[22]

If the greater number of the genes in a certain cluster have the same biological function, the probability of the distribution is high and consequently the P-value is close to 0. A threshold t is used to determine the significance of the p-value. The annotation itself means mapping a sub tree from the GO to the cluster, to find the most suitable Ontology

Sub tree (OST) that represents this cluster. The P-value is computed for each category in GO, such that the OST with the least value is the most significant one.

Mapping the TP-tree onto GO-hierarchy:

The child-parent relationship in the GO hierarchy was used to evaluate the child-parent relationship in the TP-cluster tree. So if C is a TP-cluster and C' is one of its descendent and if H is the OST of C and H' is the OST of C' then C' is biologically descendent of C if $H' = \langle$ H [22], which means the root of H' appears in H. So the SHTP algorithm starts with a node of the TP-cluster tree, and extracts the OST of this cluster, if the OST is not biologically viable or related to the parent OST, discard this node or cluster and its descendents and continue with the next sibling according to the predefined traversal order [22]. Their experiments showed that the two pruning techniques they used which are the OST and the threshold of the P-value made it a faster algorithm.

This method's goal resembles ours in terms of trying to combine domain knowledge from data with domain knowledge from literature. They do this in an intelligent way using the hyper geometric distribution. One drawback in this algorithm is that it can not be extended to large databases because the whole database has to be resident in the memory to compute the ranking function.

An Ontology-Driven Clustering Method for supporting Gene Expression Analysis [40]

The impetus of using ontologies in clustering is that classical clustering of data records lacks the ability of describing the biological meaning of similarity relationships represented [40]. The proposed approach uses similarity/distance measure on the items in the GO hierarchy. The similarity between two terms c_i, c_j in the hierarchy is computed using Lin's similarity model.

$$\mathrm{Sim}(c_i, c_j) = \frac{2 * \max_{c \in S(ci,cj)_i} (\log(p(c)))}{\log(p(ci)) + \log(p(cj))}$$

Where $S(c_i, c_j)$ are the set of parent terms shared by both c_i, c_j [40].

P(c): is the probability of finding c or one of its children in the annotation database being analyzed [40]. This algorithm was used for clustering gene records using the Gene Ontology hierarchy and it gave good results.

Ontology-Based Induction of High Level Classification Rules. [35]

In this work, a high level classification is performed by developing a query-based tool which combines knowledge encoded in ontologies and knowledge stored in relational databases to find interesting information by extending the query into a collection of queries against the database. According to this query extension, general classification rules at different levels of abstraction will be produced. Specifically, the ontology labels are augmented with frequency counts from the database to guide the classification process and prune the search space [35]. The counts are grouped by class for each concept in the hierarchy. Then, the evaluation function of the (attribute,value) pairs D(Av) is computed for each concept that is a strong indicator for class membership, and the selected pair should require the fewest additional constraints to create a query that is satisfied by tuples in single class [35]. The advantage of this algorithm is that it allows the databases to be queried at high level of abstraction [35]. One drawback of this work is that the classification process is fully dependent on the involved ontology. The accuracy of the produced classification rules depends on the correctness of the ontology hierarchy.

Wordnet improves Text Document Clustering.[13]

This work is a different topic from what we have demonstrated so far, clustering documents is the process of organizing and grouping the large amount of information into small significant collections. As the authors indicated, the previous text clustering methods are based on related terminologies between documents other than the conceptual meaning of concepts [13]. Therefore, they propose an algorithm that incorporates background domain knowledge, namely Wordnet, to overcome this deficiency. WordNet is a semantic lexicon for the English language, it groups English words into sets of synonyms called synsets, provides short, general definitions, and records the various semantic relations between these synonym sets [41]. Clearly they employ the semantic relations between the concepts while clustering the documents.

Their approach starts with composing a terms vector for each document that encodes the frequencies of the terms. Several steps of processing is done on the terms within the vectors as follows:

- Step one: Removal of the stop words (the, is, there, in, which....) from the vectors. They are meaningless.
- Step two: Stemming the words using Porter Stemmer. Basically stemming is the process of reducing a word to its stem or root form since the morphological variants of words have similar semantic interpretations.
- Step three: Pruning the rare terms using a threshold. A rare term that does not affect the clustering results. Formally, a term is exclude from the vectors representation if $\sum_{d \in D} tf\ (d, t) \leq \delta$; where tf(d,t) is the term frequency of a term in a particular document.

- Step Four: tfidf: Term frequency inverse document frequency. This measure weighs the frequency of the term in a document, with a factor that discounts its significance when it appears in almost all the documents.[13]

$$TFIDF\ (d,t) = \log(\ tf\ (d,t) + 1) * \log\left(\frac{|\ D\ |}{df\ (t)}\right)$$

The set D contains all documents, whereas df(t) is the number of documents in which the term t appears, f(d,t) is the term frequency of the document.

When the tfidf is applied, the frequency of a term in the document is replaced by the tfidf value for all terms in the vector that represents a particular document.

Compiling background knowledge into the text document representation:

Including concepts from WordNet resolves synonyms and introduces more general concepts help in identifying related concepts [13].

Concepts from WordNET are included in the terms' representation in different methods such as:

- Adding concepts C to the term vector. The concept frequency of a document d is computed by investigating all the terms in the vector against this concept. So, if a term is related to this concept in WordNet, then the frequency of the concept is increased.
- Replacing terms in the vector representation. If a term has at least one corresponding concept in WordNet. However, terms that do not appear in WordNet are not eliminated.
- Concept vector only. All terms are replaced with their corresponding concepts from WordNet and terms that do not appear in WordNet are eliminated.

The mapping of terms to concepts is challenging, since a plethora of concepts may be retrieved from WordNet and this may lead to ambiguity. It is unclear which is the most suitable concepts to include. Different ways are found to resolve this issue such as adding all concepts to the vector representation, or selecting only the first one, because originally concepts are retrieved from WordNet in a certain order that shows the most common word in English to be the first concept with the highest rank.

In their experiment, the representation was clustered using Bi-section mean clustering and the similarity between two documents $d_1, d_2 \in D$ is measured by the cosine measure between their corresponding vectors t1, t2 respectively.

$$\cos(t1,t2) = \frac{t1 \circ t2}{\| t1 \| \circ \| t2 \|}$$

The clustering results (P) were evaluated using pre-categorized clusters (L) that were done manually on this dataset. Purity of both partitions P, L is computed using the following formula:

$$Purity\ (P,L) = \sum_{p \in P} \frac{|\ p\ |}{|\ D\ |} \max_{l \in L} (prescision\ \ (p,l))$$

Precision (P,L)= $\frac{|\ p \cap l\ |}{|\ p\ |}$; measures the closeness of results between the two clusters.

The results showed that clustering with background knowledge yielded purity up to 61.8% while clustering without background knowledge yielded purity 57%, Furthermore, tfidf was a significant factor in text representation in both cases.

The advantage of this work is integrating different combinations of parameters such as disambiguation and term weighting, to enhance the clustering results. Comparing this to our work we did use the term weighting but with a different formula, which also contributed to enhancing our results.

In this chapter, we discussed literature as an important source of information, because it includes the necessary background knowledge that is used by researchers to improve the knowledge discovery process. We also discussed the two different types of background knowledge; structured and unstructured background knowledge. Then we discussed the advantages and the disadvantages of each type. After that, in Section 2.2 we talked about literature based discovery, we saw several examples of literature based discovery. Finally, we discussed the combination of background knowledge from data with literature based discovery. In the next chapter we will discuss the pattern discovery process. We will mainly focus on association rule mining since it is the data mining technique we are using in this thesis.

Chapter 3

Association Analysis

Association analysis is a methodology in data mining for discovering interesting relationships hidden in large databases. The discovered relationships are typically represented in two forms: *frequent itemsets* and *association rules*. Frequent itemsets represent subsets of items that appear together frequently in the data, e.g., {diaper, milk}; whereas association rules suggest the existence of relationships between the presence of items in the antecedent and the consequent of the rules, e.g., (cough ➔ fever).

Association rule mining was initially developed for business intelligence in which the market basket transactions are analyzed to identify potential opportunities for applications such as marketing promotions and shelf management. For example, suppose the rule (diapers ➔ beer) is discovered from the set of transactions recorded in a certain store. Store managers may act on the suggestion made by the rule by keeping both items close to each other on the shelves because they are often sold together. Association rule mining can also be applied to the medical domain to extract relationships in the clinical data. For example, it can be used as a reminder system in clinical decision support tools to remind physicians about other potential medical conditions that may explain the symptoms exhibited by a patient.

In this chapter, we present an overview of association rule mining. In Section 3.2 we discuss some of the algorithms that are currently used to efficiently extract the

association rules. Section 3.3 demonstrates some of the techniques used to identify interesting association rules and to eliminate the ones that are not useful and redundant.

3.1 Problem Definition

In this section we introduce the terminology used to explain some of the concepts in associations analysis.

Definition 1 (Items): Let I = $\{i_1, i_2, ..., i_m\}$ denote the set of items available in a data set.

Definition 2 (Itemset): An itemset X is a nonempty subset of all items, i.e., $X \subseteq I$. If an itemset contains k items then it is called k-itemset.

Definition 3 (Transactions): Let T be the set of all transactions.

Definition 4: (Support). The support count of a given itemset, denoted as σ(X), is the number of transactions that contain the particular itemset: $\sigma(X) = |\{t_i \mid X \subseteq t_i, t_i \in T\}|$, where |.| denote the number of elements in a set. The support of a given itemset is the fraction of all transactions that contain the itemset: $support(X,Y) = \frac{\sigma(X \cup Y)}{|T|}$.

Definition 5 (Frequent Itemset): A frequent itemset is an itemset whose support is greater than or equal to some user-specified minimum support threshold.

Definition 6 (Association Rule): An association rule is an implication expression of the form X➔Y, where X and Y are disjoint itemsets.

Definition 7 (Confidence): The confidence of an association rule X ➔ Y determines how frequently items in Y appear in transactions that contains items in X.

$Confidence = \frac{\sigma(X \cap Y)}{\sigma(X)}$ where σ is the predefined support count for the itemset

Definition 8: (Association Rule Discovery)

Given a set of transactions T, find all the rules that have support ≥ minsup and confidence ≥ minconf; where minsup and minconf are the corresponding support and confidence thresholds.

3.2 Association Rule Mining

As illustrated in definition 8 in Section 3.1, association rule mining is the process of finding all the association rules that pass the condition of min support and min confidence. In order to mine these rules, first the support and confidence values have to be computed for all of the rules and then compare them with the threshold values to prune the rules with low values of either support or confidence. But this process is computationally expensive because there is an exponential number of rules that can be extracted from a dataset (D) according to the following formula:

$$R = 3^d - 2^{d+1} + 1$$

Where d is the number of items in the dataset and R is the number of extracted rules.

However a large number of these rules will be pruned after applying the support and confidence thresholds. Therefore the previous computations will be wasted. To avoid this problem and to improve the performance of the rule discovery algorithm, the procedure for mining association rules is divided into two steps. The first step is frequent itemsets generation, in which all the itemsets that satisfy the minimum support threshold are extracted. The second step is rule generation, in which rules that have high confidence are formed from the previously extracted frequent itemsets. By using this methodology, the pruning step of the spurious rules is done before completing all the computations. For

example, if we have the itemset { cough, fever, cold}, the support of all the rules formed from this itemset is identical because the support of any candidate rule is the support of {cough U cold U fever} . Therefore, if this itemset is infrequent, then all candidate rules such as ({cough → cold, fever},{cold → cough, fever},{fever → cough, cold}) are guaranteed to be infrequent.

3.2.1 Frequent Itemsets Generation

Given a data set d that contains k items, the number of itemsets that could be generated is $2^k - 1$, excluding the empty set. In order to mine the frequent itemsets, the support of each itemset must be computed by scanning each transaction in the dataset. A brute force approach for doing this will be computationally expensive due to the exponential number of itemsets whose support counts must be determined.

There are several algorithms that have been proposed to improve the efficiency of frequent itemset generation. There are two typical strategies adopted by these algorithms: the first is an effective pruning strategy to reduce the combinatorial search space of candidate itemsets. The second strategy is to use a compressed data representation to facilitate in-core processing of the itemsets. Below, we give an example of each strategy.

Apriori Algorithm

The Apriori algorithm is one of the first algorithms to effectively reduce the combinatorial search space. The algorithm is guided by the concept of support-based pruning to systematically control the exponential growth of candidate itemsets. The pruning strategy, also known as the Apriori Principle, states that if an itemset is frequent then all of its subsets must also be frequent. For example, if we have a frequent itemset {cough, fever}, then any transaction that contains {cough, fever} must also contain its

subsets {cough}, {fever}. Hence {cough} and {fever} must also be frequent. Conversely, if an itemset {cough, fever} is infrequent, then all of its supersets such as {cough, fever, cold} or {cough, fever, swelling, cold} will be infrequent. The Apriori principle therefore allows us to prune the number of itemsets to be examined during frequent itemset generation.

Apriori is a breadth-first search algorithm that generates candidate itemsets of length k based on the frequent itemsets of length (k-1) found in the previous pass. It then scans the database once to determine the support counts of the candidate itemsets. Candidate itemsets whose support counts satisfy the support threshold requirement are subsequently declared as frequent itemsets.

The second strategy that is often adopted for the efficient generation of frequent itemsets is to use a compressed representation of the data. An example of this approach is the FP-Growth algorithm, which stores the data in memory using a data structure called FP-tree. This approach allows us to determine the support count by accessing the data structure in memory, instead of repeatedly scanning the transactions on the disk. In the FP-tree representation, each transaction in the database is represented as a path from the root of the FP-tree. Compression is achieved by merging prefix paths of transactions that share the same items. A vertical database representation is another popular strategy, in which each item is associated with a column of values indicating the transactions that contain the item. For example, algorithms such as CHARM [40] use vertical tid-lists as their data representation while MAFIA [41] and DCI-Closed [42] use vertical bit vectors.

In our proposed framework, we use a frequent itemset mining algorithm called PGMiner, which is a time and space efficient algorithm recently developed by our team.

Prefix Graph Miner Algorithm

Prefix-Gragh Miner (PGMiner) is a graph-based approach for mining frequent itemsets. This approach was developed by members of our team. Our approach consists of constructing a prefix graph structure and decomposing the database to variable length bit vectors assigned to nodes of the graph. The main advantage of this representation is that the bit vectors at each node are relatively shorter than those produced by existing vertical mining methods. This facilitates fast frequency counting of itemsets via intersection operations. In our method we have a condensed representation of the database in memory which is vital to achieve good efficiency. Existing algorithms such as FPGrowth [11] constructs a frequent pattern tree (FP-tree) structure to encode the relevant itemsets and frequency information. In this representation, each transaction in the database is represented as a path from the root of the FP-tree. Compression is achieved by merging prefix paths of transactions that share the same items. A vertical database representation is another popular strategy, in which each item is associated with a column of values indicating the transactions that contain the item. For example, algorithms such as *CHARM* [45] use vertical tid-lists as their data representation while *MAFIA* [6] and *DCI-Closed* [23] use vertical bit vectors.

FP-tree has been known as a good method in providing a compressed representation of the data due to the sharing items between the transactions. Nevertheless, when the database is sparse or the support threshold is low, FP-tree storage requirements may exceed the database size.

Database Name	Db. Size	Min Support	FP-Tree		Vertical Bit vector Size	PrefixGraph Size
			Num Nodes	Size		
Chess	474.4Kb	25%	31812	621Kb	19.9Kb	239.6Kb
Pumsb	14.0Mb	45%	183349	3.5MB	377.2Kb	4.28MB
WebDoc	1150.4M	10%	50313644	959.6M	52.8MB	111.6MB
Kosarak	36.8Mb	.08%	3425391	65.3MB	189.3MB	9.2MB

Table 1: Characteristics of various condensed representations[24]

Table 1 shows that the memory requirements for storing vertical bit vectors are generally less than that for FP-tree, except for the case of *Pumsb* data set. The advantage of using Vertical bit vectors is that it allow fast support counting using simple bitwise AND operations. However, when the database is large and sparse, the handling of long bit-vectors is quite inefficient since there are lots of zeros in the vectors.

Prefix Graph Representation

In this section, we describe the *PrefixGraph* representation and show in detail the construction of this structure. The items in a itemset that contains k items (k-itemset) are assumed to be sorted according to some total order π. We use the notation x π y to indicate x precedes y according to the total order. A *PrefixGraph* consists of a set of nodes and a set of directed edges connecting pairs of nodes. Any item in the database that satisfies the minimum support threshold is represented as a node in the *PrefixGraph*. Each node is also associated with a projected bit vector database (Figure 4). The following definitions are necessary concepts used during the construction of the Prefix Graph.

Definition 1 **(Prefix 2-Item)** At each node k, an item i is called its prefix 2-item if $i \pi k$ and $\{i, k\}$ is a frequent 2-itemset.

***Definition 2* (Prefix Itemset)** Consider an itemset $I = \{i_1, i_2, \ldots, i_{j-1}, i_j, i_{j+1}, \ldots, i_n\}$. A prefix itemset of I with respect to node i_j is defined as all the items $\{i_1, i_2, \ldots i_{j-1}\}$.

Prefix Graph Construction

We now illustrate the construction of the *PrefixGraph* using the transaction database given in Table 2 with support threshold $\xi = 2$. Consider the *PrefixGraph* shown in Figure 4 for the sample database in Table 2. The total order of the nodes are $a \pi\ b\ \pi\ d\ \pi\ e\ \pi\ c$. The prefix 2-item for node *b* is *a*, while the prefix 2-items for node *c* are *a* and *b*. The nodes *d, e,* and *c* are suffix nodes of *b* because *b* precedes these nodes and {*b,d*}, {*b,e*}, and {*b,c*} are frequent 2-itemsets.

Transaction ID	Items	Frequent Items
1	*a, b, c, d,*	*a, b, d, c*
2	*b, d, a, e, f, g*	*a, b, d, e*
3	*d, a, e*	*a, d, e*
4	*i, a, c, b*	*a, b, c*
5	*b, c, e*	*b, e, c*
6	*D, e, h*	*d, e*

Table 2: Sample Database[24]

First, we scan the database to identify the set of frequent items and their corresponding support counts. These frequent items are used to form the nodes of the *PrefixGraph*. For the database listed in Table 2, the list of frequent items are {(a:4), (b:4), (c:3), (d:4), (e:4)}, where the notation (m:n), *m* is the item and *n* is the support count. Now we order the nodes based on the descending order of support count. Next, we scan the database again to identify the prefix 2-items for each node. For example, the frequent 2-itemsets for nodes a, b, d, e, and c are (ab, ad, ac, ae), (ba, bd, be, bc), (da, db, de), (ea, eb, ed),

and (ca, cb), respectively. The prefix 2-items and their corresponding support counts for these nodes are { }, {a:3}, {a:3, b:2}, {a:2, b:2, d:3}, and {a:2, b:3} respectively. For each node, we store its set of prefix 2-items in a header table.

In the next step of the graph construction the transactions are stored as bits in the projected bit vector database of the nodes. We scan the database, and for each transaction, infrequent items are removed and the remaining items are sorted based on the frequency descending order. Let T be the resulting itemset. Now for each item *k* in T, we select the node *k* and compare its prefix 2-items against the prefix itemset of T. If there is a match (of at least one item), then these matching items are stored as bits in the projected bit vector database of node *k*.

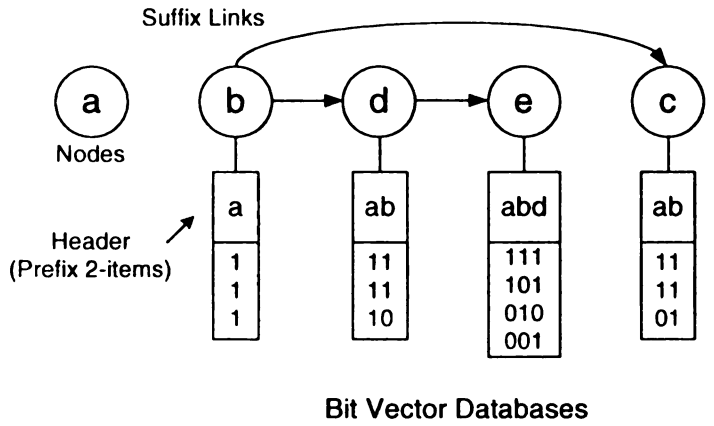

Figure 4: PrefixGraph Representation[24]

For example, consider the first transaction of the sample database. After removing the infrequent items, the remaining items are: T= {a, b, d, c}. Since the transaction has 4 frequent items we need to consider the nodes *a*, *b*, *d*, and *c*. Node *a* has no prefix 2-items and therefore nothing is stored. For node *b*, item *a* of the transaction matches with its prefix 2-item (i.e. a) and therefore bit <1> is stored in its projected bit vector DB. For Node *d*, items {a, b} of T match with its prefix 2-items and therefore bits <11> are stored

in the bit vector DB. Similarly for node *c*, the bits for items {a, b} of the transaction are stored in the bit vector databases. For the second transaction, we get T={a, b, d, e} after removing the infrequent items. Since item *a* matches with prefix 2-item of node *b*, it is stored as a bit. Items {a, b} match with prefix 2-items of node *d* and therefore they are stored as bits <11> in the database. For node *e*, items {a, b, d} of the transaction match with its prefix 2-items. So the bit vector stored in node *e* is <111>. When storing a transaction such as {d, e} at node *e*, we need to store bits <001> in node *e*, since only item *d* of the transaction matches with the prefix 2-items of node *e*.

Algorithm 1 (PrefixGraph Construction)

- Scan the database DB and find the frequent 1-itemsets (nodes) and their supports.
- Sort the nodes in the descending order of support.
- Scan DB again and find the frequent 2-itemsets for each node and create the header tables.

1. Scan the DB and do the following for each transaction:
    i. Sort the frequent items of the transaction in descending frequency order.
    ii. For each item *k* in the transaction, select node *k* and match the prefix 2-items of node *k* with the items in the transactions and if there is a match, store the matching items as a bit vector in the bit vector database of node *k*.

Analysis of PrefixGraph Structure

Based on the algorithm, we need three scans of the transaction database. The first two scans are necessary to find frequent 1-itemset and 2-itemset, while the third scan is needed to construct the projected bit vector database. If the number of items in the DB is known to be small, it is possible to combine the first two scans into one and perform the algorithm with two scans of the database.

Lemma 1: The size of the projected bit vector database of a node is bounded by the support count of the node times the number of prefix 2-items of that node.

Proof: Let m be the number of prefix 2-items of a node k. Since the number of transactions that contain item k is equal to the support count $\sigma(k)$, the size of the projected bit vector database of node k must be equal to $\lceil m \times \sigma(k)/8 \rceil$ bytes. But in a bit vector database, projected transactions starting with item k are not stored as bit vectors at that node. Therefore, the size of the projected bit vector database at node k is $\leq \lceil m \times \sigma(k)/8 \rceil$ bytes.

Lemma 1 shows an important benefit of the *PrefixGraph* structure as we use bit vector intersections during mining. According to Lemma 1, the length of the bit vector is at most equal to the support count of the node. Since the support count of an item is usually much smaller than the database size, we will have shorter bit vectors and accordingly faster bit vector intersections.

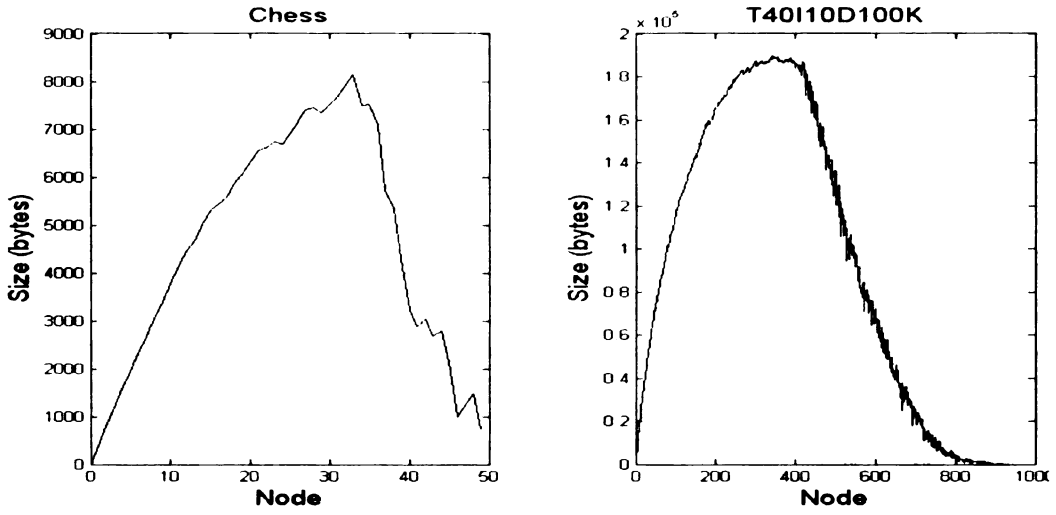

Figure 5: Size of bit vector databases at each node[24]

The size of the projected bit vector database at each node varies as shown in Figure 4 for the *Chess* and *T40I10D100K* databases with minimum support thresholds 30% and 0.10% respectively. Except for the nodes in the middle, all other nodes have small projected bit vector databases, which facilitate faster mining of itemsets.

3.2.2 Rule Generation

After generating frequent itemsets, the next step is to generate association rules from the frequent itemsets. In principle, each frequent k-itemset can generate up to $2^k - 2$ association rules. For example, rules such as (urinary ➔ kidney, nocturia), (urinary, nocturia➔kidney), etc, can be extracted from the frequent itemset {urinary, kidney, nocturia}. However, not all of these rules are interesting because they may not satisfy the minimum confidence requirements.

To efficiently generate high confidence rules from a given frequent itemset, the following confidence pruning strategy can be used. Let Y be a frequent itemset. Note that any association rule generated from Y can be written in the form of X ➔ Y – X, where X

is a subset of Y. It is possible to show that if the rule X➔Y-X does not satisfy the confidence threshold, then any rule $\overline{X}$ ➔ Y-$\overline{X}$, where $\overline{X} \subset X$ must not satisfy the confidence threshold as well. The Apriori rule generation algorithm uses this pruning strategy to efficiently generate high confidence rules from a given frequent itemset. This algorithm uses a level-wise approach for generating association rules where each level corresponds to the number of items that belong to the rule consequent [34]. For example if {abc➔d} is a low-confidence rule then all the candidate rules that have d as an item in the consequent side should be pruned. In other words, the rules {ab➔cd},{ac➔bd}, {bc➔ad},{c➔abd} should be discarded because their confidence is guaranteed to be low.

In this section we discussed how confidence and support are used for pruning the rules. In the next section we will point out additional techniques that will be used to further prune the rules.

3.2.3 Pruning Redundant Rules

After applying the support and confidence measures to prune the insignificant rules, there still some redundant rules that need to be pruned. In our system, we consider a rule X to be redundant if it has a confidence that is lower than or equal to the confidence of a related rule Y where the set of items that form the left-hand side of rule Y is a subset of the set of items that form the left-hand side of rule X:

X: A ➔ B is a rule with confidence C_1

Y: C ➔ B is a rule with confidence C_2 where C is a subset of A,

then rule X is to be pruned if $C_1 <= C_2$.

For example if rule (cough, ear pain ➔ common cold) has a confidence of 70% and the rule (cough ➔ common cold) has a confidence of 75%, then we prune the rule (cough, ear pain ➔ common cold) because knowing that a patient suffers from ear pain does not improve the confidence of predicting common cold among patients who complain about coughing.

3.3 Interesting Measures

Association rule mining algorithms may produce a large number of rules. As a result it is important to distinguish between interesting rules from spurious ones. There are various interest measures that have been developed for this purpose. These measures can be classified into two types: objective versus subjective interestingness measures. An objective measure uses statistics computed from the data to determine whether a rule is interesting. Examples of the objective measures are support, confidence, and the lift measure. In the previous section we mentioned how the support and confidence were used to prune unimportant rules.

In contrast, subjective interestingness measures tend to require domain knowledge to determine whether a rule is interesting. In general, a rule is considered to be subjectively interesting if it reveals unexpected information about the data or is potentially actionable. For example, the rule cough ➔ cold is not subjectively interesting despite its high support and confidence values because it is obvious to most physicians. On the other hand, the rule fish oil ➔ Raynaud disease is interesting because it associates a substance that may be used to treat a disease. In this thesis we will investigate the use of

background knowledge from literature to determine the interestingness of rules extracted from the medical domain.

3.4 Summary

In this chapter we described the association rule mining task, including the frequent itemsets generation and rule generation processes. We also discussed the pruning approach used to eliminate redundant rules. Finally, we explained the use of objective and subjective interestingness measures for pruning spurious and uninteresting rules. In the next chapter we show how these approaches form the key components of our proposed framework.

Chapter 4

Proposed Framework

The main goal of this thesis is to discover interesting rules in large databases by incorporating background knowledge from literature. This chapter describes our proposed methodology in the context of mining association rules from the medical domain. More specifically, we integrate the rules extracted from an EMR database with domain knowledge automatically acquired from a large repository of biomedical literature. We then show how the background knowledge helps to distinguish rules that are obvious to the domain experts from those that are either spurious or probable.

The remainder of this chapter is organized as follows, in Section 4.1 we introduce the various components of our framework. Section 4.2 describes how the background knowledge from biomedical literature can be used to provide a subjective way for evaluating and interpreting the rules.

4.1 MIR: Mining Interesting Rules from Medical Data

MIR, which stands for Mining Interesting Rules, is a web-based system we had developed for discovering interesting relationships in the medical domain. MIR can be used as a recommendation tool to support clinical decision making. It may also be used for research purposes because some of the interesting rules may reveal previously unknown relationships that need to be further investigated. The overall system was built using the C# .NET framework.

Figure 6 illustrates the overall architecture of our proposed system. The input data for our system corresponds to an electronic medical records (EMR) database, which contains both demographic and clinical information about the patients (including their medical complaints, medication, age, and insurance information). The database records all the activities corresponding to 1,145,517 clinical visits by 427,214 patients between 2001 and 2005.

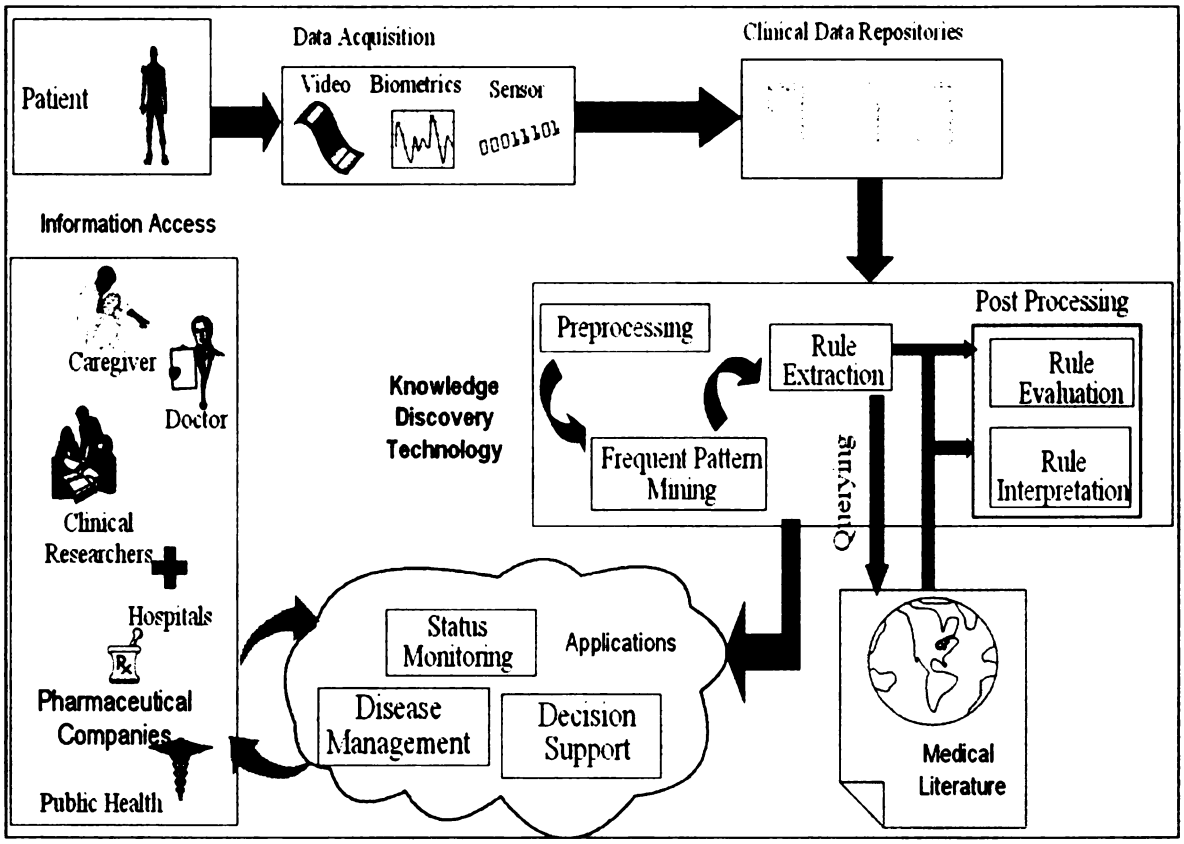

Figure 6: Overall Framework

The proposed framework consists of 3 major components: preprocessing, data mining and post-processing. Since the original EMR data is stored in a relational database that contains 23 tables, it has to be transformed into a binary transaction format so that it can be subsequently mined by current association analysis algorithms. For the EMR data set, we are interested in examining the relationships between various symptoms and diseases

of patients. The information was originally stored in a table describing the medical complaints of patients. After preprocessing, a transaction data file is created where each transaction corresponds to a patient's visit to her healthcare provider. Each transaction also contains a list of "items" that represent the symptoms and diseases of the patient.

The data mining component performs three major tasks. First, the PG-Miner algorithm described in Section 3.2.1 is applied to generate frequent itemsets from the transaction database. Next, the Apriori rule generation algorithm is applied to extract high confidence rules from the frequent itemsets. Finally, the rule pruning step described in Section 3.2.3 is used to eliminate redundant rules.

Rules that survived the support, confidence, and redundancy pruning steps are subsequently evaluated and interpreted during the post-processing step using background knowledge acquired from the literature. A detailed description of the rule evaluation and interpretation steps is described in the next section.

4.2 Rule Evaluation and Interpretation

The goal of the post-processing component of our framework is to perform the following two tasks:

- *Rule Evaluation.* In this step the rules are evaluated using background knowledge from literature in order to prune those that are subjectively uninteresting.
- *Rule interpretation*: In this step the background knowledge is used to aid the interpretation of the extracted rules.

Note that the rule evaluation and interpretation tasks are not exactly independent. This is because the ability to interpret the meaning of a rule also gives us an indication whether it

is interesting or potentially spurious. Therefore we explain the steps for evaluating and interpreting the rules together in the remainder of this chapter.

Our framework uses Pubmed as the source for background knowledge. PubMed is a bibliographic database developed at the National Library of Medicine to provide access to citations from biomedical literature. The procedure of querying and retrieving background knowledge from Pubmed is described in Section 4.2.1.

The information retrieved from PubMed is used in the following two contexts. First, it is used to distinguish between *obvious* rules from *unexpected* ones. A rule (A → C) is considered obvious if there is strong evidence in the literature that A and C is associated with each other. Otherwise, the rule is said to be unexpected. We describe our method for distinguishing obvious rules from unexpected rules in Section 4.2.2. Second, the background knowledge is also used to distinguish two kinds of unexpected rules: *spurious* rules and *probable* rules. An unexpected rule (A → C) is considered probable if there is sufficient evidence in the literature to suggest that the concepts A and C are indirectly related via some other intermediate concepts. Otherwise, the unexpected rule is considered spurious. We describe our approach for distinguishing probable rules from spurious rules in Section 4.2.3.

4.2.1 Querying Background Knowledge from Pubmed

As previously mentioned, our proposed framework uses Pubmed as the source of background knowledge. To perform rule evaluation and interpretation, our proposed framework supports two types of queries to PubMed:

1. A *support-based query* to determine the frequency of occurrences of the queried terms in the literature.
2. A *retrieval query* to find all the abstracts that contain the queried terms.

A set of e-utilities is available for download from the National Library of Medicine (NLM) web site at http://eutils.ncbi.nlm.nih.gov to facilitate easy access to PubMed. We had written a C# program to access PubMed using the e-utilities. Our program seeds the query with the search terms along with their corresponding parameters and then submits the query to the Entrez PubMed search engine. There are two steps involved for retrieving information from PubMed: *search* and *fetch*. The search step is executed using the E-search utility while the fetch step is executed using the E-fetch utility. For support-based queries, it is sufficient to use the E-search utility to determine the frequency of occurrences for the queried terms. For retrieval queries, we need to execute both E-search and E-fetch utilities.

E-search Utility

The general format for querying PubMed using the e-search utility is given as follows:

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?$param_1$=val_1& ...&$param_n$=val_n)

where $param_i$ is a search parameter and val_i is its corresponding value. Below is a list of the parameters to be passed to the E-search query:

- URL of the NLM website which is (http://eutils.ncbi.nlm.nih.gov).
- Type of e-utilities operation to be executed. Since this is a search process, we request for the E-search utility.
- db parameter, which is the name of the database.

- Usehistory parameter. If this parameter is set to "yes" the history of this search is stored for subsequent retrievals.
- The search term, which is the queried word or phrase.

For example if we want to search for the term "diabetes mellitus" the following query can be used:

(http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=Pubmed&usehistory=y&term=diabetes%20mellitus)

E-fetch Utility

The fetch step is invoked to retrieve all the abstracts previously located by the search step. Each search query has its own history, with different information stored in the history. The queryno and the webenv are the most important values used to retrieve records associated with the search query results. The format of an E-fetch query is given as follows:

(http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi? $param_1=val_1\&...\&param_n=val_n$)

The following search parameters can be used in the E-fetch query:

- URL of the NLM website which is (http://eutils.ncbi.nlm.nih.gov).
- Type of the e-utilities operation to be executed. Since this is a fetch process, we request for the E-fetch utility.
- db parameter, which is the name of the database.
- Retmax parameter, which is the maximum number of records to be retrieved.
- Querykey parameter, which is a numeric value that indexes the records retrieved by the E-search query.
- Webenv parameter, which is a string value that indexes the E-search query.

- Retmode parameter, which: specifies the mode in which the records are retrieved; for example, xml or ASCII text. In our case we set this parameter to xml and then parse the xml file to extract the individual abstracts.
- Rettype parameter, which specifies the type in which the records are retrieved; for example abstract or brief summary. In our case we set the rettype to be abstract.

The following is an example of a fetch query statement:

(http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?&db=Pubmed&WebEnv=" + "32165HJ4JH7980EIUR345231" + "&query_key=" +1+ "&retmode=xml&rettype= abstract" + &retmax=" + 1000).

Each query (Esearch and Efetch) incurs a certain cost in terms of execution time. The required time for query execution depends on the number of records retrieved per query. PubMed imposes an upper limit to the number of abstracts that can be fetched in a single query. Therefore, if a queried term appears in more than 1000 abstracts, we need to invoke the e-fetch query multiple times. In Chapter 5, we showed that it is possible to reduce the number of queries to PubMed without losing important information.

Once the abstracts have been retrieved, we need to apply the followng text preprocessing steps to extract the relevant background information:

1. Tokenization, which is the process of converting a stream of characters in the abstract into a stream of words (terms).
2. Stopword removal, which is the process of discarding general terms that are not useful for the rest of the mining task.

3. Stemming, which is the process of stripping the suffixes of words in the abstract.
4. Support counting, which is the process of counting the frequency of occurrences for each of the words that appear in the retrieved abstracts.

Data Dictionary

The data dictionary is a data structure we had created to store the key medical terms when processing the abstracts retrieved from PubMed. The data dictionary is created based on all the words describing the "items" that exist in the EMR transaction file. The purpose of this dictionary is to eliminate irrelevant terms and stopwords that exist in the PubMed abstracts. We use the data dictionary when the abstracts are tokenized. Each tokenized word is checked against the dictionary to examine whether it is a relevant medical term or not. If it is available in the dictionary, we will count the frequency of the word. Otherwise the term is ignored and the parser moves to the next term.

Porter Stemmer

The Porter stemming algorithm is used to remove the morphological endings of words. Stemming helps to normalize the terms that have common stems but different endings. For example, the terms allergic, allergy, and allergies all share a common stem but with different suffixes. The stemmer is invoked after tokenizing the abstract and removing the stopwords.

4.2.2 Distinguishing Obvious Rules from Unexpected Rules

Objective measures such as support and confidence only help to remove rules that are statistically insignificant. Some of the rules that survive the support, confidence, and

redundancy pruning steps may still not be interesting. For example, consider the rule cough ➔ cold, which has relatively high support and confidence values. Such a rule is still considered uninteresting because the relationship is obvious and well known to the domain experts. To eliminate such obvious rules, we need to incorporate subjective knowledge into the rule evaluation and interpretation step. Throughout this thesis, we call the non-obvious rules as unexpected rules.

MIR uses background knowledge from the Pubmed repository to distinguish the obvious rules from unexpected ones. We assume that the support count of a rule in the PubMed literature provides a good estimate for the obviousness of the rule. The higher the literature support count, the more obvious the rule is. To distinguish between the support of a rule in the EMR data from its support in the PubMed literature, we use the term *expected support* throughout the thesis referring to the support obtained from PubMed literature.

Therefore, our framework classifies the subjective interestingness of a rule as follows: a rule is classified as obvious if the expected support is greater than or equals to a user-specified minimum threshold; otherwise, it is classified as unexpected. To perform such classification, we use the support-based query described in Section 4.2.1 to determine the co-occurrence frequency of the rule in Pubmed abstracts. Identifying the appropriate expected support threshold is not an easy task. We investigated several candidate thresholds, such as using the mean, median or 80^{th} percentile of the expected support for all the rules. We observe experimentally that the median seems to yield the best results, so we set out threshold to the median value.

4.2.3 Distinguishing Probable Rules from Spurious Rules

After removing the obvious rules, the remaining rule set contains only unexpected rules, i.e., rules that are non-obvious because they involve only concepts that are rarely associated together in the literature. These unexpected rules may represent two types of relationships. First, the relationship suggested by the rule may still be spurious despite satisfying the support and confidence thresholds. Second, a hidden (indirect) relationship may exist between the concepts in the rule antecedent and consequent. Rules with such hidden relationship are known as probable rules. We explain our methodology for distinguishing probable rules from spurious rules in the rest of this section.

Given a rule A → C, our objective is to build a graph that relates the concepts on each side of the rule (A and C). If such a graph can be established, then the rule is deemed to be probable. The intermediate nodes in the graph represent the common terms extracted from the abstracts retrieved from PubMed when querying the concepts A and C separately. However, instead of using all the words that appear in the retrieved abstracts, we consider only the top n highest frequency terms as intermediate nodes. Note that this approach is similar to the literature-based discovery process described in Chapter 2. However, unlike traditional literature-based discovery, our framework is more general because the intermediate term B does not have to be directly associated with both A and C. There may be more than one intermediate node along the path between A and C. Put another way, we say that the interaction between A and C involves more than one level of indirection.

This approach has the danger of relating two seemingly unrelated concepts A and C using several levels of indirection. In this situation, the intermediate nodes of the graph

may correspond to general terms that relate to most of the items in the database. We consider two approaches to avoid this problem. First, we check the data dictionary to avoid terms that are too general for the domain (e.g., disease). Second, we restrict the maximum number of indirection levels allowed by the framework. If the number of indirection levels is greater than some threshold (say, 2), the rule is classified as spurious.

We treat the task of constructing an association graph between concepts A and C as a path discovery problem. We explain how MIR addresses this problem in the next section.

Path Discovery Problem

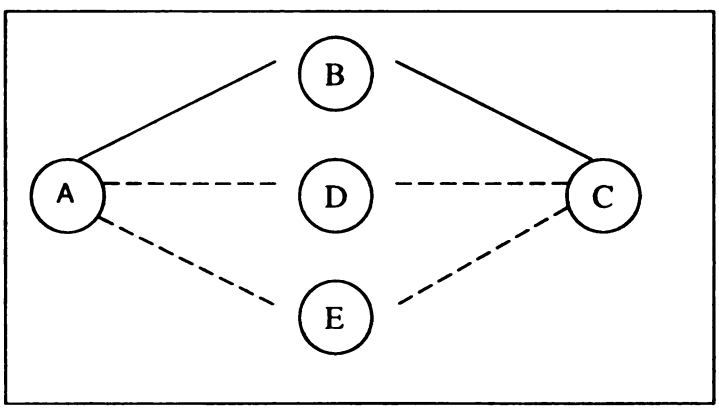

Figure7: 1-level network between concepts of a rule's sides.

Let G = (V, E) be a weighted graph of vertices (V) and edges (E). Each vertex corresponds to one of the concepts in the data dictionary. An edge between two nodes P and Q in the association graph indicates that Q must belong in the top *n* list of highest frequency terms associated with P. The weight of an edge (P, Q) represents the expected support between the concepts P and Q.

Given a source node (A $\in$ V) and a destination node (C $\in$ V), the path discovery problem aims to discover a path connecting between nodes A and C (see Figure 7). Unlike traditional path discovery problems, the weights of the edges between nodes are

initially unknown and therefore must be queried using PubMed. Since each query incurs some cost, our goal is to construct the graph without incurring too much overhead for querying PubMed.

To do this, we start with a bidirectional search by querying the concepts on each side of the rule simultaneously (say, A and C). After retrieving the abstracts corresponding to A and C, we determine the top-n highest frequency terms associated with A and C. If there is at least one common term between A and C, the task is complete and the rule is declared as probable. Otherwise, if there is no common terms that are strongly associated with both A and C, we proceed to the next level and query PubMed again. To make the process more efficient, we should not have to query all the terms that belong to the top-*n* list for A and C separately. Instead, we will choose only the most promising node to expand (i.e., to be queried) at the next level of indirection. We apply a utility function to determine whether a node is a promising candidate for expansion. For example, let $\{B_1, B_2, \ldots, B_n\}$ denote the list of top-*n* terms associated with the concept A. For each candidate node B_i, its utility function is given as follows:

$$\text{Utility Function} = \lambda * u(A, B_i) + (1 - \lambda) * \bar{u}(B_i, C)$$

where $u(A,B_i)$ is the expected support obtained from PubMed literature and $\bar{u}(B_i, C)$ corresponds to the expected support between B_i and C. Since $\bar{u}(B_i, C)$ is unknown before querying, we estimate the value using the actual support for itemset $\{B_i, C\}$ observed in the EMR database. However, the support for $\{B_i, C\}$ was already recorded when applying the PGMiner algorithm to generate frequent itemsets.

We also consider an alternative approach for finding intermediate nodes using MESH terms. MESH (Medical Subject Headings) is a controlled vocabulary thesaurus provided by the National Library of Medicine.

Table 3: Algorithm for identifying probable and spurious rules using MeSH.

Input: A set of unexpected rules {A→C}

Output: Sets of probable and spurious rules

1. Execute an automated query for the consequent and antecedent of the rule, in order to retrieve the relevant abstracts .
2. **for each** concept ∈ (A, C)
3. Parse the abstracts and distinguish the medical terms using the dictionary.
4. Compute the document frequency (DF) for each term.
5. Compute the significance of the terms according to the Utility Function value.
6. Sort the terms according to their frequency values.
7. Select the top-n terms and map each term to its corresponding MeSH concept(s).
8. **end**
9. Intersect the top n-list of the MeSH concepts for both sides of the rule.
10. If iteration number < 2 and intersection set is nonempty, then add the rule to the probable rule set.
11. If iteration number < 2 and intersection set is empty then for both sides on the established network, choose the term with the highest utility value and goto step1.
12. If iteration number = 2 add the rule to spurious rule set.

Instead of intersecting the top-n highest frequency terms to identify the intermediate nodes, we first mapped the terms in the top-n lists to their corresponding MESH concepts. We then intersect the list of MESH concepts to identify whether there are any common MESH terms that can be used as intermediate nodes of the association graph. The MESH-based approach is summarized in Table 3.

Figure 8 illustrates an example of how MESH concepts can be used to find intermediary nodes for distinguishing spurious rules from probable rules.

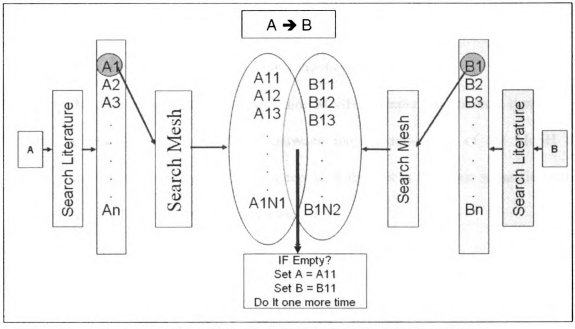

Figure 8: An illustration of using MeSH concepts for rule evaluation.

4.2.4 Rule Evaluation and Interpretation using UMLS Ontology

Our proposed framework also allows us to apply the UMLS ontology to evaluate and interpret the rules. UMLS encodes the high-level medical concepts (called semantic types) as well as semantic relations between them. Each MeSH concept is first mapped to its corresponding semantic type(s) and to the semantic relations participated by their corresponding semantic types. The semantic relations provide additional background

Table 4: Algorithm for identifying probable and spurious rules using MeSH and UMLS.

Input: Set of unexpected rules A→B

Output: Sets of Probable / spurious rules

1. **for each** rule A→B
2. Get the corresponding MeSH concepts and semantic types for A and B.
3. **for each** concept, d ∈ {A,B}
   4. Query PubMed and retrieve the abstracts corresponding to the concept d.
   5. Compute the document frequency of each term that appears in the abstracts.
   5. Compute the utility function of the terms (see Section 4.2.3).
   6. Sort the terms according to their frequency values.
   7. Map the terms to their corresponding MeSH concepts and semantic types.
   8. Check for semantic relation between the semantic type of the MeSH term and the semantic type of the concept d. If there is at least one semantic relation, add the MeSH term to the top *n* list.
9. **end**
10. Intersect the top n-list of the MeSH concepts for both sides of the rule.
11. If iteration number < 2 and intersection set is nonempty, then add the rule to the probable rule set.
12. If iteration number < 2 and intersection set is empty then for both sides on the established network, choose the term with the highest utility value and goto step1.
13. If iteration number = 2 add the rule to spurious rule set.

knowledge that can be used to avoid false alarms (i.e., spurious edges in the association graph) resulting from using only the MeSH concepts or top-n terms when finding indirect relationships between concepts. The UMLS ontology imposes additional constraints that

must be satisfied by the terms to ensure that only terms that are semantically related to the search query terms (on each rule side) are selected as intermediate nodes for the association graph. As a result MeSH concepts that are unrelated to the queried terms will be pruned and this will reduce the number of the false positives in the output.

Table 4 summarizes the outline of the UMLS based rule evaluation and interpretation procedure. An illustration of this procedure is also shown in Figure 9. Note that all the rules classified as probable according to this procedure are potentially interesting and worth further investigation by the domain expert because they suggest a hidden relation between the two concepts in the antecedent and consequent of the rule. These hidden relations are not obvious and may suggest new knowledge that was previously unknown to the experts. Using UMLS, interpreting these relations will not be too difficult because the semantic relations provide additional information to guide the experts to understand the meaning of the association graph.

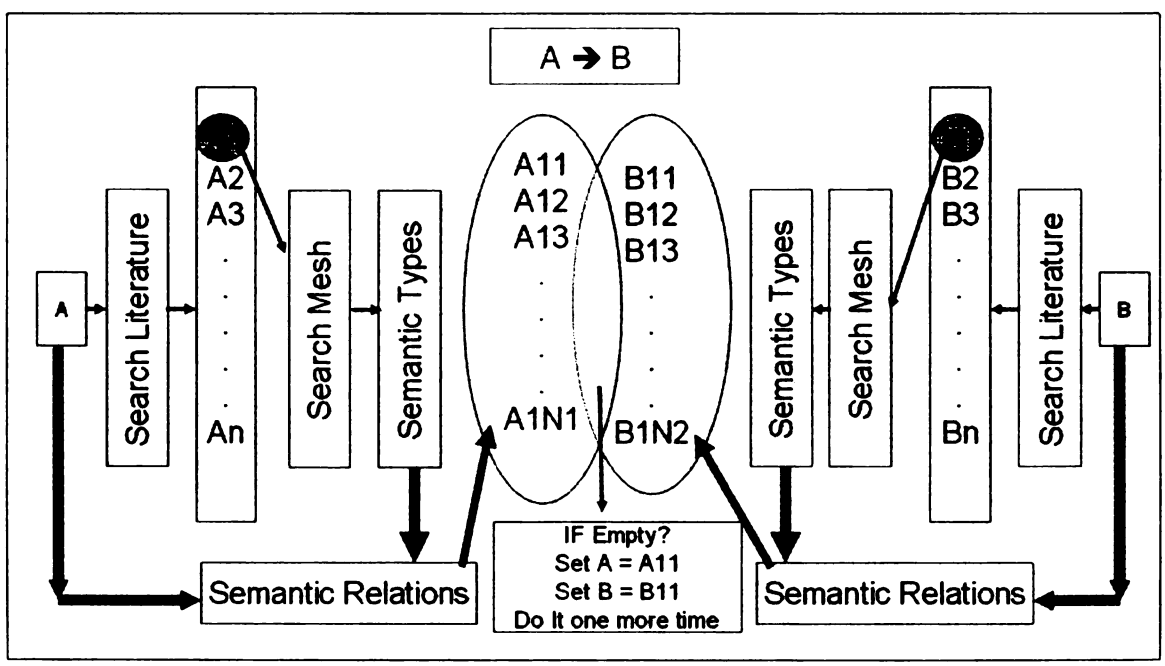

Figure 9: An illustration of using UMLS for rule evaluation and interpretation.

4.3 Summary

This chapter presents an overall framework of the MIR system. We present the various components of the system and describe the methods used for incorporating background knowledge from PubMed and UMLS. In the next chapter we will demonstrate the experiments we had conducted in order to evaluate the effectiveness of the methods described in this chapter.

Chapter 5

Experiments

This chapter demonstrates the experiments that we have performed to evaluate the various components of our system. We use the MQIC EMR data, which has been deidentified by the GE Healthcare data Consortium, for our experiments. The EMR data contains 1,145,517 clinical visit records of 427,214 patients during the period of 2001-2005. The EMR is stored in a relational database that contains 23 tables such as patients, complaints, medications, insurance, etc. the background knowledge for the system is acquired from two sources: Pubmed, which is a repository of biomedical abstracts, and UMLS (Unified Medical Language System).

Our experiments will be conducted to achieve the following objectives:

- To show the efficiency (in terms of runtime and storage requirements) of the PGMiner frequent itemset generation algorithm.
- To determine the reliability of estimating support counts from the Pubmed literature.
- To determine the effectiveness of distinguishing obvious rules from unexcpected rules using support counts from Pubmed.
- To determine the effectiveness of distinguishing probable rules from spurious rules using Pubmed and MeSH ontology..
- To interpret the rules using UMLS.

5.1 Experiment 1: Frequent Itemsets Generation

In this experiment, we want to compare the efficiency of using the PGMiner algorithm against other existing frequent itemsets generating algorithms (Apriori and FPGrowth). The input to the algorithms is a file that contains 3115630 transactions and 5819 items. Each transaction corresponds to a patient's visit to the health care provider and each item corresponds to a symptom or disease. As described in Section 3.1, the definition of a frequent itemset depends on the support threshold. Changing the support threshold affects the number of frequent itemsets generated. The higher the support threshold is, the fewer frequent itemsets produced by the algorithm. Figure 10 and Table 5 shows the effect of applying different support thresholds to the transaction data used in our experiments.

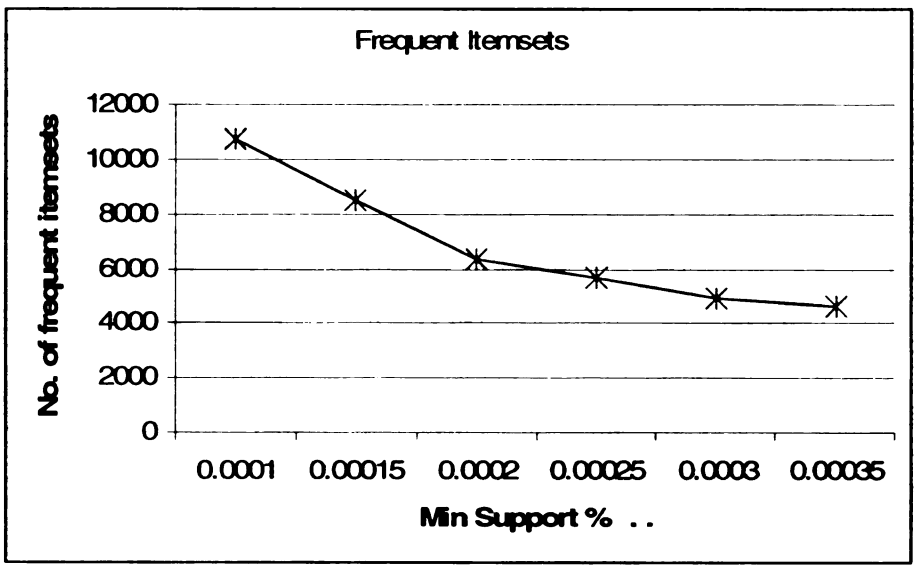

Figure 10: frequent itemsets and min support

min support %	0.0001	0.00015	0.0002	0.00025	0.0003	0.00035
Itemsets	10728	8486	6302	5700	4886	4585

Table 5: Generated itemsets using different support thresholds.

For instance, when the support value is .0001% the number of frequent itemsets generated is 10728 while the number of frequent itemsets in the case of .00035% is 4585.

As described in Section 3.2.1, the Apriori frequent itemsets generation algorithm uses support-based pruning to efficiently discover frequent itemsets. Despite its efficiency in terms of pruning the exponential search space, the algorithm is still quite slow because it needs to make multiple passes over the database. Figure 11 and Table 6 shows a runtime comparison between PGMiner and Apriori.

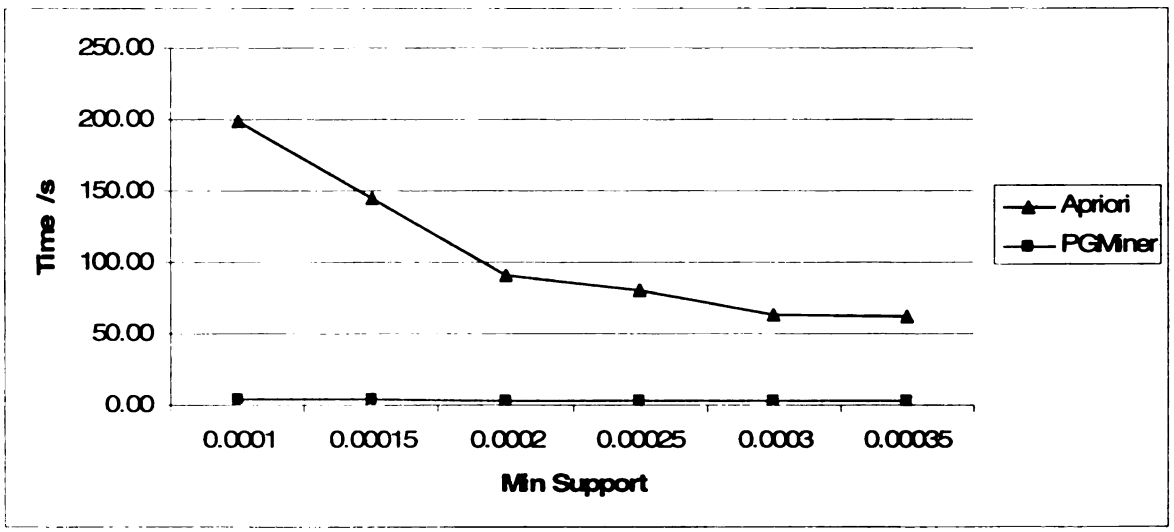

Figure 11: comparison between PGMiner and Apriori

Min Support %	0.0001	0.00015	0.0002	0.00025	0.0003	0.00035
Apriori /sec	199.02	145.00	90.16	80.02	63.78	62.41
PGMiner /sec	3.39	3.34	3.20	3.22	3.20	3.13

Table 6: Runtime comparison between Apriori and PGMiner

Notice that the execution time for PGMiner is extremely fast compared to the execution time for Apriori. For instance, when the support threshold was set to .0002%, the runtime for Apriori was 90.16 seconds while PGMiner finished running in 3.20 seconds, which is about 30 times improvement. Furthermore decreasing the support threshold does not affect the execution time for PGMiner by much because the mining step is performed almost instantaneously for this range of support thresholds. In contrast, the Apriori algorithm shows an exponential growth in its execution time when decreasing the support threshold. For example, when using the support .00035%, Apriori needed 62.41 seconds

to generate the itemsets compared to PGMiner which needed only 3.13 seconds to generate the same frequent itemsets. When the support threshold is further reduced to .0001%, the Apriori algorithm increased to 199.02 seconds while PGMiner increased by only .26 seconds.

The bottleneck of Apriori has to do with the number of passes it makes on the transaction database. To circumvent this problem, the FPGrowth algorithm was proposed by Pei et al. [11] to reduce the I/O overhead. FPGrowth constructs a condensed representation of the transaction data in memory so that subsequent mining steps can be performed efficiently without going to the disk multiple times. However, the tradeoff here is that FPGrowth consumes a large amount of memory to store the data and thus may not be scalable when the database becomes very large (relative to the size of main memory). PGMiner also constructs an internal representation of the transaction database, called PrefixGraph. Figure 12 shows a comparison between the memory requirements of PGMiner and FPGrowth algorithm. It is evidently clear that PGMiner is more space efficient than FPGrowth.

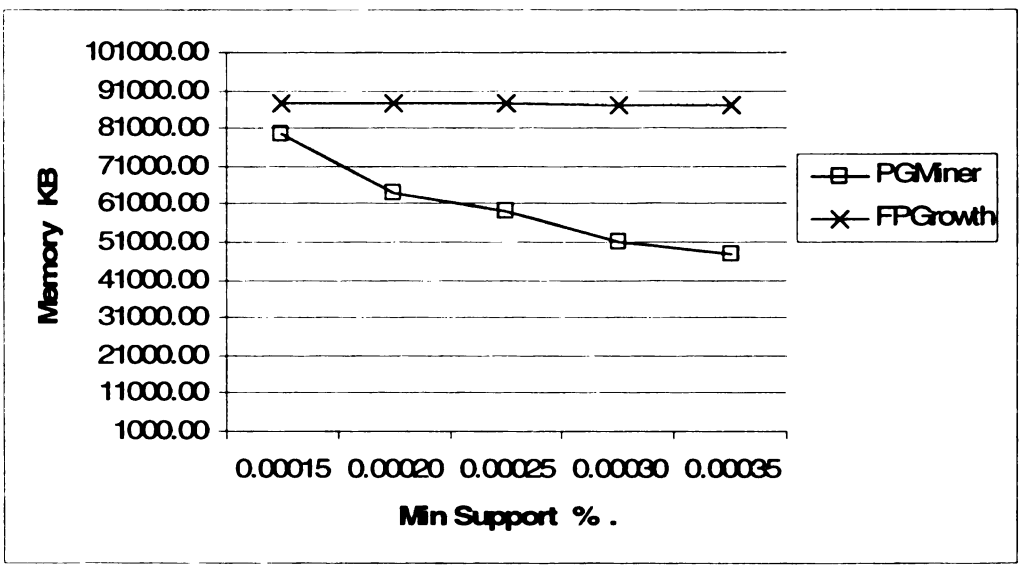

Figure 12: Comparison between PGMiner and FPgrowth in memory requirements

For example, when the support threshold is set to .0002% the memory consumed by PGMiner is about 60000KB whereas FPGrowth uses 90000KB. Furthermore, from our experience, the runtime of PGMiner is less than the runtime of FPGrowth.

After frequent itemsets generation, we applied Apriori rule generation algorithm to generate the association rules. As noted in Section 3.2.2 the confidence metric is used to prune spurious rules. Rules that do not pass the minimum confidence requirement will be pruned. Initially, the number of extracted rules depends on the support threshold because the support threshold determines the number of frequent itemsets produced. We have conducted an experiment to show the effect of the varying the support threshold on the number of extracted rules. We fixed the confidence threshold to be 65%. Fig.13 shows the increasing number of rules generated when the support threshold is decreased.

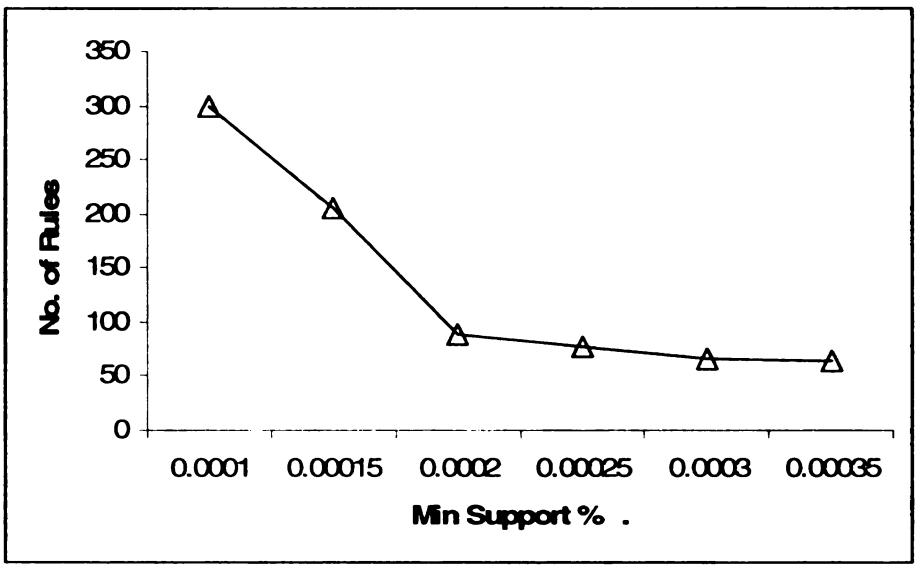

Figure 13: Number of rules versus Min Support

In Figure 14 we performed another experiment to show the effect of changing confidence on the number of rules generated. For this experiment, the support threshold is fixed at .00015%. Notice that when the confidence value is increased the number of the rules decreases quite substantially.

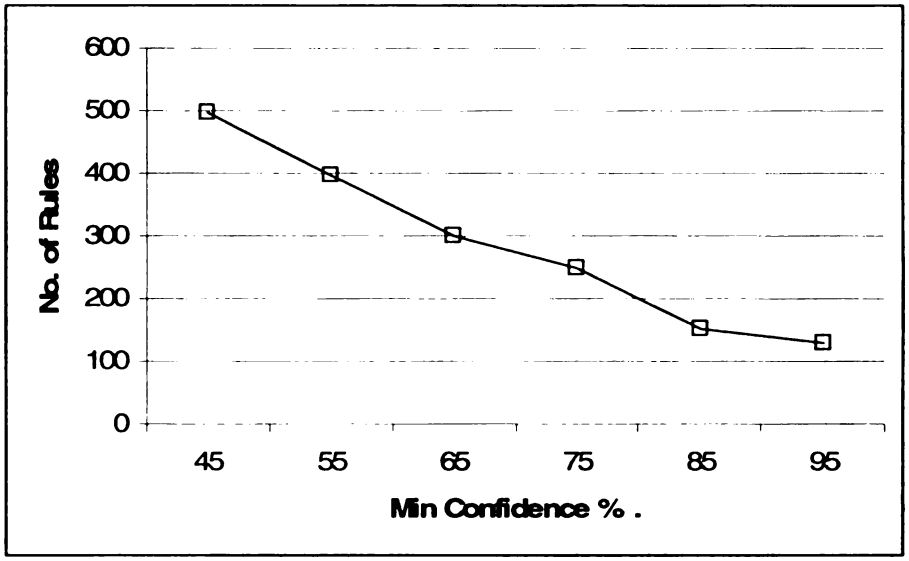

Figure 14: Number of Rules versus Confidence

In spite of this, the resulting rule set still contains some redundant rules. The pruning approach described in Section 3.2.3 is then applied to eliminate redundant rules that have confidence less than or equal to their corresponding parent rules.

5.2 Experiment 2: Estimating the Support from Pubmed

Pubmed is a huge repository of biomedical literature. Information is accessible through a search query using search terms such as title words, author names, phrases, words, journal names. As described in Section 4.2.2, one of the key steps in evaluating the subjective interestingness of a rule (A→C) generated from the EMR database is to identify the top-n terms that appear frequently with both A and C. These high frequency terms may serve as intermediary concepts to help better explain the relationship between A and C. However, in order to identify the top-n terms for A (or C), we need to query Pubmed and retrieve all the abstracts that contain the word A (or C). In some cases, the number of abstracts associated with a queried term may exceed 1000, which is the maximum number of abstracts retrieved for a given query from Pubmed. To retrieve

more abstracts, say 4000, we need to query Pubmed multiple times, to fetch the abstracts number 1-1000, 1001-2000, 2001-3000, and so on. This makes the querying process becomes highly inefficient. To improve its efficiency, we investigate the possibility of determining the top-n terms associated with a given concept using a sample of the abstracts. The challenge here is to determine the appropriate sample size so that the top-n terms determined from the sample is identical to the top-n terms determined from all the relevant abstracts. We conducted the following experiment to determine the appropriate sample size. We queried the term Dysplasia and found 39009 occurrences of the term in Pubmed. Our goal is to find the top-15 terms that appear frequently with the term Dysplasia by varying the sample size from 1000 to 5000, 10000, 20000, and 39009 (using all the abstracts). Figure 15 shows the results of this experiment.

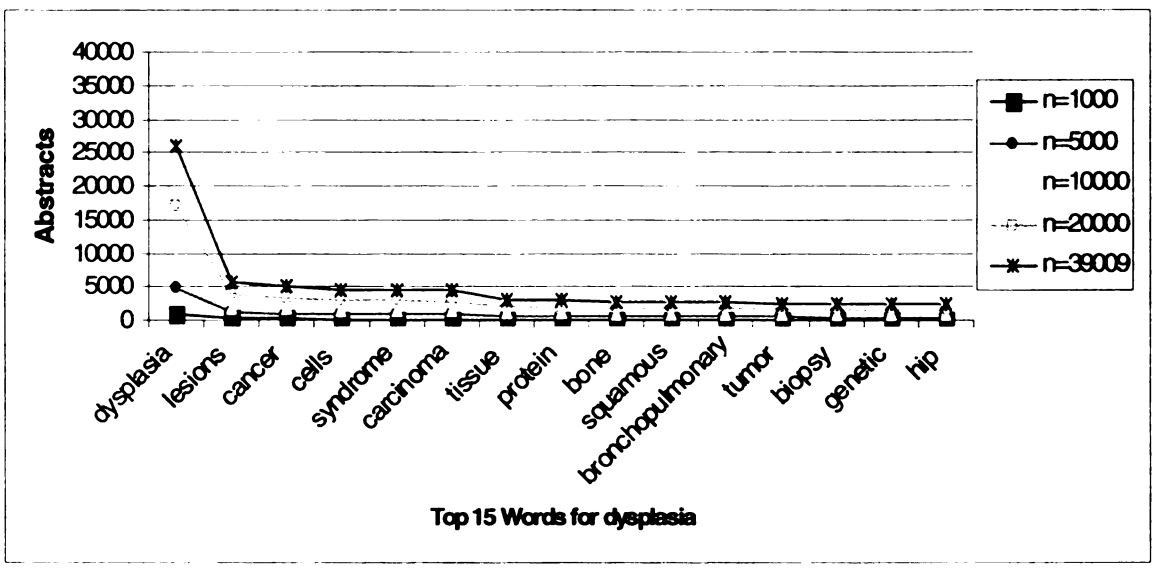

Figure 15: Document Frequency of terms in Pubmed

The experimental results show that the order of the top-15 words is consistent with each other, irrespective of the sample size. This result suggests that it is possible to obtain a good estimate of the top-15 terms that appear frequently with a given concept based on a sample of 1000 abstracts alone (i.e., in a single query).

5.3 Experiment 3: Distinguishing Obvious Rules from Unexpected Rules

We want to show in this experiment the role in which Pubmed plays in filtering obvious rules from unexpected ones. We first extracted the rules with high support (more than .00015%) and high confidence (more than 65%) from the EMR data using PGMiner and the Apriori rule generation algorithm. There were 113 rules that pass the minimum support and minimum confidence thresholds. For each association rule A→C in the rule set we query Pubmed to find the number of abstracts that contain both concepts A and C. The frequency of {A,C} represents the expected support of the rule based on background knowledge.

We argue that if the expected support of the rule is high, then the extracted rule should be obvious because the relationship between A and C is prevalent in the literature. On the other hand, if the expected support of the rule is low (less than a specified support threshold), the rule is considered unexpected because A and C are seldom observed together in literature. In our experiment, we set the expected support threshold to be the median value of the expected support for all the extracted rules. Since the median value was 13, any rule that has an expected support greater than or equals to 13 is considered an obvious rule, while the rest are declared as unexpected rules. To validate these results, we consulted with a physician from the Michigan State University Health Center. We requested him to classify the 113 rules into two groups (obvious versus unexpected). Table 7 shows the confusion matrix of the result. Notice that our framework could distinguish 48 obvious rules out of 57 described as obvious by the physician.

		Predicted	
		Obvious	**Unexpected**
Actual	**Obvious**	48	9
	Unexpected	19	37

Table 7: Confusion Matrix

The overall accuracy of our framework is 75.2%.

Notice that there are 19 rules declared as obvious by our framework but marked as unexpected according to the physician. This is because the expected support computed from literature may not be indicative of causal relations between two medical concepts. For example, the rule osteomalacia ➔ osteomyelitis is classified as unexpected by the physician because there is no known relation between both diseases. However, since they are both diseases related to the bone (osteomyelitis is an acute bone infection while osteomalacia is the softening of the bones caused by the deficiency of vitamin D), they often appear together in the literature as examples of possible diagnosis for a bone-related medical condition. In addition, 9 of the rules declared as unexpected by our framework are marked as obvious by the physicians. This is because some of the relationships are so obvious that they are rarely mentioned together in the literature (e.g., urinary frequency slowing ➔ nocturia).

The following are some examples of rules that were classified correctly by our framework. The rule (swelling ➔ leg edema) is an obvious association because the value of the support from literature is (1506) which is higher than the literature-based support threshold, 13. Examples of other obvious rules that were classified correctly include (obesity hyperglycemia ➔ hypercholesterolemia), (hematuria urinary incontinence➔ nocturia). An example of an unexpected association that was identified by our method and the physician is (obesity Grave's disease ==> hypertension) which is classified as

unexpected since its frequency in Pubmed is 2 which is less than the expected support threshold (13).

Applying the frequencies from Pubmed literature alone does not help in determining whether the unexpected association is spurious or interesting. Therefore in the next section we describe how our system addresses this issue.

5.4 Experiment 4: Distinguishing Probable Rules from Spurious Rules

The previous section describes how to use the expected support from Pubmed to distinguish between obvious and unexpected rules. A rule is declared to be unexpected if its expected support is low because there is lack of evidence in the literature suggesting that the concepts are directly related. Having a low expected support however does not necessarily mean that the rule is spurious. There may still exist a hidden (indirect) relation between the given concepts via some intermediary concepts. We consider rules that are indirectly associated via hidden concepts as probable rules.

Unlike the closed literature based discovery approach described in Section 2.2, we allow the concepts to be associated at more than one level of indirection. For example, the relation between A and C may involve two levels of indirection:

$$A \leftrightarrow B_1 \leftrightarrow D \leftrightarrow B_2 \leftrightarrow C$$

In general the more levels of indirection allowed by the framework, the more likely A and C become indirectly associated. However, we observed that if the number of indirection levels is too large, the intermediate concepts became overly general, thus relating two concepts that might not have any association between them. To avoid this

problem, we set the maximum level of indirection to be 2, i.e., any unexpected rule that requires more than 2 levels of indirection is classified as spurious. Otherwise, the unexpected rule is classified as probable. Note that the intermediate concepts can be extracted based on either the top-15 words that appear frequently with a given concept or mesh terms mapped from the top-15 words, (see the discussion in Section 4.2.3). To evaluate these results, we again consult with the physician from the Michigan State University Health Center and requested him to categorize his previously declared unexpected rules into two groups: spurious versus probable. Table 8 shows the confusion matrix of the results using top-15 terms.

		Predicted		
		Obvious	**Probable**	**Spurious**
Actual	**Obvious**	48	9	0
	Probable	11	7	2
	Spurious	8	23	5

Table 8: Confusion Matrix using terms

Our framework managed to correctly identify 7 out of the 9 probable rules classified by the physician. On the other hand, we managed to correctly classify only 5 out of the 28 spurious rules. The poor result can be explained by the following reasons. First our framework is too conservative in terms of classifying a rule as spurious, to ensure that it does not miss any interesting rules. This problem can be avoided using a stricter criterion for finding intermediary concepts (e.g. using the top-5 terms instead of top-15 terms). Second, the concepts implied by the rules may be indirectly associated via general medical terms such as cell, tissue, or syndrome. To avoid this problem, we need to remove these general terms from the data dictionary we had used when processing Pubmed abstracts.

When evaluating the unexpected rules using MeSH concepts, we obtained similar results as before. Table 9 shows the confusion matrix of the classification results.

		Predicted		
		Obvious	**Probable**	**Spurious**
Actual	**Obvious**	48	9	0
	Probable	11	8	1
	Spurious	8	23	5

Table 9: Confusion Matrix using MeSH

The following are some examples of unexpected rules that were classified correctly by our framework. We observed that the rule (benign prostatic hypertrophy Peyronie's disease ➔ nocturia) was correctly identified as spurious. This is because we do not find any intermediate terms that can relate to both sides of the rule, neither in the first nor second level of indirection. The rule (urinary frequency, prostate nodule ➔ benign prostatic hypertrophy) is an example of a probable rule since we could locate the intermediary term (tumor). The expected support of the rule is equal to 0.

Finally, we used a well-known example from literature based discovery to test the effectiveness of our system. Our objective is to make sure that MIR is capable of finding indirect relations that are well known in the literature. As previously mentioned in Section 2.2.1, Swanson found a relationship between Raynaud's disease and fish oil through blood viscosity and platelet aggregation. As will be shown here, we were able to reproduce this result by using MeSH concepts as intermediate terms. We found blood viscosity and blood aggregation to be in the top 15 MeSH terms for both Raynaud's disease and fish oil. We reached to this result by querying both sides of the rule and computing the frequencies of all the related medical terms that appear in the abstracts (e.g., terms such as fat, oil, lipid, and blood for fish oil and terms such as cold, skin,

blood, and temperature for Raynaud's disease). Table (8) shows the complete list of top 15 terms extracted from the abstracts associated with each side of the rule.

Fish Oil	DF	Raynaud's disease	DF
acid	566	Syndrome	232
Oil	388	Scterosis	196
Diet	284	Vascular	165
Lipid	209	Cold	141
Cells	160	Skin	138
Plasma	145	Blood	133
Blood	128	Scleroderma	133
Fat	127	Tissue	124
Protein	127	Hand	98
inflammatory	114	Antibodies	93
cardiovascular	101	Crest	84
metabolism	91	Finger	80
Tissue	90	Flow	78
Heart	90	Temperature	68
Serum	86	Doppler	68

Table 8: List of top15 for Fish Oil→Raynaud's Disease

We then sort the list according to their frequency values and map the top 15 terms in each list to their corresponding MeSH concepts. Finally we intersect the MeSH concepts and found blood viscosity and platelet aggregation to be the intermediate MeSH concepts between Raynaud's disease and fish oil.

5.5 Experiment 5: Rule Interpretation using UMLS

In this experiment, we continue to use the Swanson example described in Experiment 4 to explain how the structured knowledge encoded in UMLS can be used for rule interpretation. Using UMLS, we first extract the semantic types of the concepts for both sides of the rule. We then query PubMed to find the candidate intermediate terms. Next, we map each intermediate term to its MeSH concept and corresponding semantic types.

Finally, we verify that a semantic relation exists in UMLS between the semantic type of the intermediate MeSH concept and the semantic types of the concepts corresponding to each side of the rule.

Once we capture at least one semantic relation, we add the MeSH concept to the top-15 list of the intermediate terms. For example, the word (blood) was mapped to the MeSH concept (blood viscosity) and this concept has the semantic type *Physiologic Function*. The semantic type *Physiologic Function* is related to fish oil (which has the semantic type *Biologically Active Substance*) through the semantic relations *disrupts*, *complicates*, and *affects*. As a result, blood viscosity is added to the top 15 list of fish oil. Similarly, for Raynaud's disease (which has a semantic type *Disease or Syndrome*), we found the semantic relations *process*, *manifestation-of*, *affects*, and *result-of* between its semantic type and the semantic type for blood viscosity (*Physiologic Function*). As a result blood viscosity is also added to the top-15 list for Raynaud's disease. After intersecting the top-15 lists, we found a common intermediate MeSH concept, which is blood viscosity.

The interpretation of the result is illustrated in Figure 16. It shows the relations between the antecedent of the rule (fish oil) and the intermediate term Blood viscosity through the relation type (affects). On the other hand, we can find the consequent of the rule (Raynaud's disease) related to blood viscosity by the association (causes). This background knowledge that we included from UMLS simplified the task of understanding the rule.

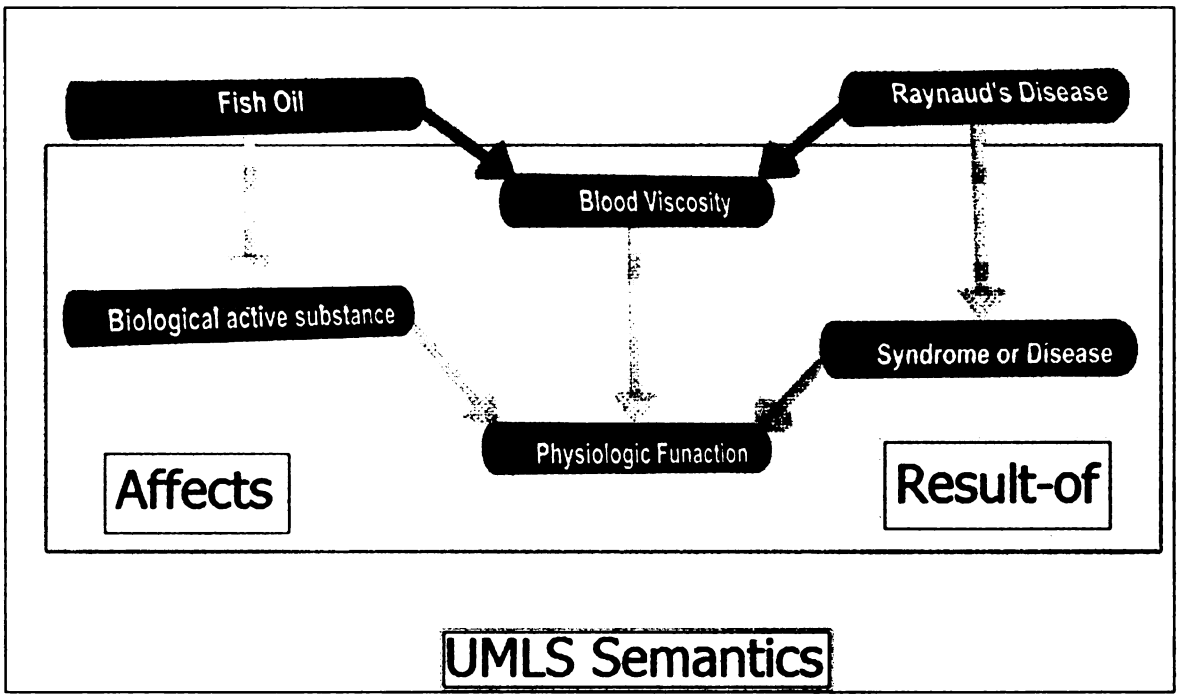

Figure 16 – An illustrative example of UMLS

5.6 Summary

In this chapter we have performed several experiments to evaluate the various components in our system. First, we showed that PGMiner is more efficient than other algorithms of frequent itemsets generation. Second, we could estimate the support counts from Pubmed literature. After that, we could determine the effectiveness of our algorithm in distinguishing obvious rules from unexpected ones. Moreover, we were able to effectively exclude interesting from spurious rules in the set of unexpected rules. and finally, we managed to use the background knowledge in UMLS to evaluate and interpret the interesting rules.

Chapter 6

Conclusion and Future Work

The main objective of our thesis is to investigate the problem of mining interesting association rules from large databases in the presence of background knowledge. Towards this end, we develop a framework called MIR for finding interesting association rules in the medical domain by incorporating background knowledge from PubMed and UMLS. We showed that the structured (UMLS) and unstructured (PubMed) background knowledge helps to achieve a two-fold goal: (1) to distinguish obvious rules from spurious or probable ones and (2) to aid in the interpretation of the rules. We evaluated the effectiveness of our rule evaluation approaches by comparing the predictions made by the framework against the actual rule classification provided by a medical expert from the Michigan State University Health Care Center.

First, our experimental results suggest that, using background knowledge, our proposed framework may effectively distinguish obvious rules from unexpected rules. Obvious rules correspond to the associations that are highly supported in the literature while unexpected associations correspond to relationships that are observed frequently in the data but are weakly supported in the literature. Among the unexpected rules, some of them may be spurious, while others may be interesting because there are hidden concepts indirectly associated with the terms. We also showed that our proposed framework is capable of reproducing well-known results from literature-based discovery.

We have also investigated the advantages of using structured background knowledge (from UMLS). Our results indicate that the structured knowledge is not that

helpful for our data set in terms of pruning insignificant rules. This is because the predefined semantic types in UMLS are too general (e.g., terms such as physiological function, disease or syndrome, and biologically active substance). Since all the items in the data are either disease or symptoms, a semantic relation almost always exist between most of the low-level concepts investigated in this study.

We also conducted experiments to assess the efficiency of our framework in terms of generating frequent itemsets and reducing the number of queries needed for building the association graph. First, we showed that our proposed frequent itemset generation algorithm (PGMiner) is more efficient, in terms of its memory consumption and runtime, compared to other existing frequent itemset generation algorithms (Apriori and FP-growth. To improve the efficiency of querying PubMed, we showed that it is possible to obtain a good estimate of the top-n terms by retrieving only 1000 abstracts in a single query instead of making multiple queries to retrieve all the abstracts for a given concept.

Our current experience from this research highlighted some of the important issues that we believe, if handled, will enhance the performance of our work. First, we plan to use more effective methods (e.g., using natural language processing techniques) to identify the important terms (and phrases) associated with a given abstract. To do this, we may employ tools such as the NP-Parser package. It takes text as an input and chunks the text into words or simple phrases. Such phrases, when cleaned up, are useful to match to entries from controlled vocabularies such as the Medical Subject Heading (MeSH) vocabulary.

Another issue that we would like to improve is the method for integrating the supportive frequencies from Pubmed with the observed support in the data. In spite of our

early success in using the approximated frequencies, we would like to find a more reliable probabilistic approach for combining evidence from the data with evidence from literature. Furthermore, our experimental results seem to suggest that our framework tends to be more biased toward classifying a rule as probable than spurious. We believe that this occurs when choosing the top *n* terms. We plan to do more extensive research to determine the appropriate way for selecting the top *n* terms. For example, we may apply the lift measure (instead of support). This measure computes the closeness of the query search term and the candidate terms, we want to include only the terms that are conceptually more close to the query search term in our top *n* list. Another possibility is to decrease the size of the top *n* list and observe how this will affect the result.

As we mentioned earlier, using UMLS as structured background knowledge did not help in the pruning criteria, because of the generality of the semantic types. We would like to do more research in this direction and investigate for other ways for utilizing the predefined semantic network in UMLS in order to aid in the pruning process and thus identifying useful rules from spurious ones.

Finally, our current framework used the association rule mining method from data mining in order to combine the background knowledge in literature with the data from the medical domain. As part of our future work, we would like to extend the framework to other data mining tasks such as sequential pattern mining, clustering, classification, and anomaly detection.

REFERENCES

[1] Agarwal, R. C. (1998). A tree projection algorithm for generation of large itemsets for association rules. *IBM TJ Watson Research Center.*

[2] Agarwal, R. C., Aggarwal, C. C., & Prasad, V. V. V. (2001). A tree projection algorithm for generation of frequent item sets. *Journal of Parallel and Distributed Computing, 61*(3), 350-371.

[3] Agrawal, R., Imieliński, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. *ACM SIGMOD Record, 22*(2), 207-216.

[4] Agrawal, R., Shafer, J., Center, I. B. M. A. R., & San Jose, C. (1996). Parallel mining of association rules. *Knowledge and Data Engineering, IEEE Transactions on, 8*(6), 962-969.

[5] Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proc.20th Int.Conf.very Large Data Bases, VLDB,* , 487–499.

[6] Burdick, D., Calimlim, M., & Gehrke, J. (2001). Mafia: A maximal frequent itemset algorithm for transactional databases. *Proceedings of the 17th International Conference on Data Engineering,* , 443–452.

[7] Cimiano, P., & Staab, S. (2004). Learning by googling. *ACM SIGKDD Explorations Newsletter, 6*(2), 24-33.

[8] Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge discovery in databases: An overview. *AI Magazine, 13*(3), 213-228.

[9] Ganiz, M. C., Pottenger, W. M., & Janneck, C. D. Recent advances in literature based discovery.

[10] Grahne, G., & Zhu, J. (2003). Efficiently using prefix-trees in mining frequent itemsets. *Proceedings of the ICDM Workshop on Frequent Itemset Mining Implementations,*

[11] Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM SIGMOD Record, 29*(2), 1-12.

[12] Hand, D. J. (1998). Data mining: Statistics and more? *The American Statistician, 52*(2)

[13] Hotho, A., Staab, S., & Stumme, G. (2003). Wordnet improves text document clustering. *Proc.of the SIGIR 2003 Semantic Web Workshop,*

[14] Hristovski, D., Stare, J., Peterlin, B., & Dzeroski, S. (2001). Supporting discovery in medicine by association rule mining in medline and UMLS. *Medinfo, 10*(2), 1344–1348.

[15] Hristovski, D., Peterlin, B., Mitchell, J. A., & Humphrey, S. M. (2003). Improving literature based discovery support by genetic knowledge integration. *Studies in Health Technology and Informatics, 95*, 68-73.

[16] Hu, X. (2005). Mining novel connections from large online digital library using biomedical ontologies. *Library Management, 26*(4/5), 261-270.

[17] Hu, X., Yoo, I., Song, M., Zhang, Y., & Song, I. Y. (2005). Mining undiscovered public knowledge from complementary and non-interactive biomedical literature through semantic pruning. *Proceedings of the 14th ACM International Conference on Information and Knowledge Management,* , 249-250.

[18] Huang, W. (2005). Mining scientific literature to predict new relationships. *Intelligent Data Analysis, 9*(2), 219-234.

[19] Liu, B., Hsu, W., Mun, L. F., & Lee, H. Y. (1999). Finding interesting patterns using user expectations. *Knowledge and Data Engineering, IEEE Transactions on, 11*(6), 817-832.

[20] Liu, J., Paulsen, S., Wang, W., Nobel, A., & Prins, J. (2005). Mining approximate frequent itemset from noisy data. *Proceedings of the Fifth IEEE International Conference on Data Mining,* , 721-724.

[21] Liu, J., Wang, W., & Yang, J. (2004). A framework for ontology-driven subspace clustering. *Proceedings of the 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* , 623-628.

[22] Liu, J., Wang, W., & Yang, J. (2004). Gene ontology friendly biclustering of expression profiles. *Computational Systems Bioinformatics Conference, 2004.CSB 2004.Proceedings.2004 IEEE,* , 436-447.

[23] Lucchese, C., Orlando, S., & Perego, R. (2006). Fast and memory efficient mining of frequent closed itemsets. *Knowledge and Data Engineering, IEEE Transactions on, 18*(1), 21-36.

[24] Moonesinghe, K., & Tan, P. Fodeh S. (2006). Frequent closed itemset mining using prefix graphs with efficient flow-based pruning strategy.

[25] Mukherjea, S., & Sahay, S. Discovering Biomedical Relations Utilizing the World Wide Web.

[26] Nagypal, G. Improving information retrieval effectiveness by using domain knowledge stored in ontologies. *Lecture Notes in Computer Science,* , 780-789.

[27] Orlando, S., Lucchese, C., Palmerini, P., Perego, R., & Silvestri, F. (2003). kDCI: A multi-strategy algorithm for mining frequent sets. *Proc.of the Workshop on Frequent Itemset Mining Implementations, in Conjunction with ICDM 2003,*

[28] Pei, J., Han, J., & Mao, R. (2000). CLOSET: An efficient algorithm for mining frequent closed itemsets. *Proc.2000 ACM-SIGMOD Int.Workshop Data Mining and Knowledge Discovery (DMKD'00),* , 11–20.

[29] Shannon, C. E., & Weaver, W. (1963). Mathematical theory of communication. *University of Illinois Press.*

[30] Srinivasan, P. (2004). Text mining: Generating hypotheses from MEDLINE. *Journal of the American Society for Information Science and Technology, 55*(5), 396-413.

[31] Srinivasan, P., & Sehgal, A. K. Mining MEDLINE for similar genes and similar drugs. *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA,*

[32] Swanson, D., Smalheiser, N., & Torvik, V. Ranking indirect connections in literature-based discovery: The role of medical subject headings (MeSH). *Journal of the American Society for Information Science and Technology,*

[33] Swanson, D. R. (1990). Medical literature as a potential source of new knowledge. *Bulletin of the Medical Library Association, 78*(1), 29-37.

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

[35] Taylor, M., Stoffel, K., & Hendler, J. (1997). Ontology-based induction of high level classification rules. *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery,*

[36] The Knowledge Management Network, Website (2006), www.brint.com/km/.

[37] Tiffin, N., Kelso, J. F., Powell, A. R., Pan, H., Bajic, V. B., & Hide, W. A. (2005). Integration of text-and data-mining using ontologies successfully selects disease gene candidates. *Nucleic Acids Research, 33*(5), 1544-1552.

[38] Van der Eijk, C.C., van Mulligen, E. M., Kors, J. A., Mons, B., & van den Berg, J. (2004). Constructing an associative concept space for literature-based discovery.

Journal of the American Society for Information Science and Technology, 55(5), 436-444.

[39] Vos, R. (2001). Using concepts in literature-based discovery: Simulating Swanson's Raynaud–Fish oil and Migraine–Magnesium discoveries. *Journal of the American Society for Information Science and Technology, 52*(7), 548-557.

[40] Wang, H., Azuaje, F., & Bodenreider, O. (2005). An ontology-driven clustering method for supporting gene expression analysis. *Computer-Based Medical Systems, 2005.Proceedings.18th IEEE Symposium on,* , 389-394.

[41] Wikipedia. Website. (2006). www.wikipedia.org.

[42] Wren, J. D. (2004). Extending the mutual information measure to rank inferred literature relationships. *BMC Bioinformatics [Computer File], 5*, 145.

[43] Xiaohua, H., Xiaodan, Z., Illhoi, Y., & Yanqing, Z. (2006). A semantic approach for mining hidden links from complementary and non-interactive biomedical literature. *Proceedings of the Sixth SIAM International Conferance on Data Mining,*

[44] Xiaoming Chen, Xuan Zhou, Richard Scherl, and James Geller. 2.3.12 using an interest ontology for improved support in rule mining.

[45] Zaki, M. J., & Hsiao, C. J. (2002). CHARM: An efficient algorithm for closed itemset mining. *Proceedings of the Second SIAM International Conference on Data Mining,*

[46] Zhang, J., Silvescu, A., & Honavar, V. (2002). Ontology-driven induction of decision trees at multiple levels of abstraction. *Proceedings of Symposium on Abstraction, Reformulation, and Approximation,* , 316-323.