# Semantic Annotation of Frequent Patterns

QIAOZHU MEI, DONG XIN, HONG CHENG, JIAWEI HAN,
and CHENGXIANG ZHAI
University of Illinois at Urbana-Champaign

11

Using frequent patterns to analyze data has been one of the fundamental approaches in many data mining applications. Research in frequent pattern mining has so far mostly focused on developing efficient algorithms to discover various kinds of frequent patterns, but little attention has been paid to the important next step—interpreting the discovered frequent patterns. Although the compression and summarization of frequent patterns has been studied in some recent work, the proposed techniques there can only annotate a frequent pattern with nonsemantical information (e.g., support), which provides only limited help for a user to understand the patterns.

In this article, we study the novel problem of generating semantic annotations for frequent patterns. The goal is to discover the hidden meanings of a frequent pattern by annotating it with in-depth, concise, and structured information. We propose a general approach to generate such an annotation for a frequent pattern by constructing its context model, selecting informative context indicators, and extracting representative transactions and semantically similar patterns. This general approach can well incorporate the user's prior knowledge, and has potentially many applications, such as generating a dictionary-like description for a pattern, finding synonym patterns, discovering semantic relations, and summarizing semantic classes of a set of frequent patterns. Experiments on different datasets show that our approach is effective in generating semantic pattern annotations.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms: Algorithms

Additional Key Words and Phrases: Frequent pattern, pattern annotation, pattern context, pattern semantic analysis

**ACM Reference Format:**

Mei, Q., Xin, D., Cheng, H., Han, J., and Zhai, C. 2007. Semantic annotation of frequent patterns. ACM Trans. Knowl. Discov. Data, 1, 3, Article 11 (December 2007), 30 pages. DOI = 10.1145/1297332.1297335 http://doi.acm.org/10.1145/1297332.1297335

This work was supported in part by the National Science Foundation under award numbers 0425852, 0515813, and 0513678.

Authors' addresses: Q. Mei (contact author), D. Xin, H. Cheng, J. Han, and C. Zhai, Department of Computer Science, University of Illinois at Urbana-Champaign, 1334 Siebel Center, 201 N. Goodwin M/C 258, Urbana, IL 61801; email: qmeiz@uiuc.edu.

## 1. INTRODUCTION

Discovering frequent patterns from large databases has broad applications, such as association rule mining [Agrawal et al. 1993], correlation analysis [Brin et al. 1997], classification [Deshpande et al. 2003], and clustering [Wang et al. 1999]. It has been a central research topic in data mining for many years, and various techniques have been developed for mining specific frequent patterns, including frequent itemsets [Agrawal et al. 1993; Han et al. 2004], sequential patterns [Agrawal and Srikant 1995], and graph patterns [Yan and Han 2002]. These techniques can usually discover efficiently a large, complete set of frequent patterns and provide basic statistic information such as support for each pattern.

Unfortunately, the excessive volume of the output patterns and the lack of context information have made it difficult to interpret and explore the patterns. Indeed, in most cases, a user only wants to explore a small set of most interesting patterns; thus, before exploring them, it would be beneficial to give a user some rough idea about the hidden meanings of the discovered patterns and why they are interesting. This is analogous to literature reviewing. Before deciding whether to read through a paper, a reader often wants to first look at a short summary of its main ideas. Similarly, it is also highly desirable to have such a summary for a frequent pattern to suggest its potential meanings and to help a user decide whether and how to explore the pattern. Therefore, researchers have raised a new major challenge in frequent pattern mining, which is how to present and interpret the patterns discovered to best support the exploration and analysis of individual patterns. To meet this challenge and facilitate pattern interpretation, we need to annotate each frequent pattern with semantically enriched, in-depth descriptions of the pattern and its associated context.

Researchers have employed concepts like closed frequent pattern [Pasquier et al. 1999] and maximal frequent pattern [Bayardo 1998] to shrink the size of output patterns and provide more information beyond "support." Recently, novel methods have been proposed to mine a compressed set of frequent patterns [Xin et al. 2005] and to summarize a large set of patterns with the most representative ones [Yan et al. 2005], generating additional information (i.e., transaction coverage [Xin et al. 2005] and pattern profiles [Yan et al. 2005]) going beyond that of support. These methods can successfully reduce the number of output patterns and selectively present only the most interesting ones to the user. There has been another line of research which explores formal notions of statistical significance to extract significant patterns [DuMouchel and Pregibon 2001; Gionis et al. 2006; Webb 2007]. However, the information that these methods use to annotate a frequent patten is restricted to morphological information or numerical statistics (e.g., support, transaction coverage, significance, and profile). From such an annotation, users could not infer the semantics of a pattern, thus would still have to look through all the data transactions containing the pattern in order to figure out whether it is worth exploring.

In this article, we study the problem of automatically generating semantic annotations for frequent patterns, by which we mean to extract and provide

a concise and in-depth semantic for a frequent pattern that can suggest the hidden meanings of the pattern.

*What is an appropriate semantic annotation for a frequent pattern?* In general, the hidden meaning of a pattern can be inferred from patterns with similar meanings, the data objects cooccurring with it, and transactions in which the pattern appears. Annotations with such information are analogous to dictionary entries, which can be regarded as annotating each term with some structured semantic information. Consider the following example.

*Example* 1 (*Dictionary Entry*).[1]

**Dictionary Term: "*pattern*"**
['pætan], noun, . . .
**definitions:**
1) a form or model proposed for imitation
2) a natural or chance configuration
3) . . .
**example sentences:**
1) . . . a dressmaker's *pattern* . . .
2) . . . the *pattern* of events . . .
**synonym or thesaurus:**
model, archetype, design, exemplar, motif, etc.

In Example 1, we see that in a typical dictionary entry, the annotation for a term is structured as follows. First, some basic nonsemantic information is presented. Second, a group of definitions is given which suggest the semantics of the term, followed by several example sentences which show the usage of this term in context. In addition, a set of synonyms, thesaurus, or semantically similar terms are presented which have similar definitions to this term.

Analogously, if we can extract similar types of semantic information for a frequent pattern and provide such structured annotations to a user, it will be very helpful for him/her to interpret the meanings of the pattern and to further explore it. Given a frequent pattern, it is trivial to generate nonsemantic information such as basic statistics and morphological information, so the main challenge is to generate the semantic descriptions, which is the goal of our work.

Following the dictionary analogy, we have several tasks: First, we should ideally provide precise semantic definitions for a pattern like those in a dictionary. Unfortunately, this is not practical without expertise of the domain. Thus, we opt to look for information that can indicate the semantics of a frequent pattern, which presumably can help a user to infer the precise semantics. Our idea is inspired from natural language processing, where the semantics of a word can be inferred from its context, and words sharing similar contexts tend to be semantically similar [Lin and Pantel 2001]. Specifically, by defining and analyzing the context of a pattern, we can find strong *context indicators* and

[1]The example is selected from Merriam-Webster's *Collegiate Dictionary* and *Thesaurus*.

use them to represent the meanings of a pattern. Second, we want to extract the data transactions that best represent the meanings of the pattern, which is analogous to the example sentences. Finally, semantically similar patterns (SSPs) of the given pattern, namely, patterns with similar contexts to the original pattern, can be extracted and presented. This is similar to the synonyms or thesauri of a term in a dictionary.

The following is an example of such a semantic pattern annotation (SPA).

*Example* 2 (*Annotating a Frequent Pattern*).

**Pattern: "{*frequent, pattern*}"**
sequential pattern; support = 0.1%; closed
**context indicators:**
"mining", "constraint", "Apriori", "FP-growth"
"rakesh agrawal", "jiawei han", . . .
**representative transactions:**
1) mining *frequent patterns* without candidate . . .
2) . . . mining closed *frequent* graph *patterns*
**semantically similar patterns:**
"{frequent, sequential, pattern}", "{graph, pattern}"
"{maximal, pattern}", "{frequent, close, pattern}", . . .

The pattern "{frequent, pattern}" to be annotated in this example is either a frequent itemset or frequent sequential pattern in text. This dictionary-like annotation provides semantic information related to "{frequent, pattern}", consisting of its strongest context indicators, the most representative data transactions, and the most semantically similar patterns. The context indicators and representative transactions provide a view of the context of the pattern from different angles to help a user understand the pattern, while the semantically similar patterns provide a more direct connection between the pattern and any other pattern(s) already known to the user.

Despite its importance, to the best of our knowledge, the semantic annotation of frequent patterns has not been well addressed in existing work. In this work, we define the novel problem of generating semantic annotations for frequent patterns. We propose a general approach to automatically generate structured annotations as shown in Example 2, by: (1) formally defining and modeling the context of a pattern; (2) weighting context indicators based on their strength to indicate pattern semantics; and (3) ranking transactions and semantically similar patterns based on context similarity analysis. Empirical results on three different datasets show that our algorithm is effective for generating semantic pattern annotations and can be applied to various real-world tasks.

The semantic annotations generated by our algorithm have potentially many other applications, such as ranking patterns, categorizing and clustering patterns with semantics, and summarizing databases. Applications of the proposed pattern context model and semantical analysis method are also not limited to pattern annotation; other example applications include pattern

compression, transaction clustering, pattern relations discovery, and pattern synonym discovery.

The rest of the article is organized as follows. In Section 2, we formally define the problem of semantic pattern annotation and a series of its associated problems. In Section 3, we introduce how the pattern context is modeled and instantiated. Pattern semantic analysis and annotation generation is presented in Section 4. In Section 5, we further discuss how we can incorporate a user's prior knowledge into the annotation process. We discuss our experiments and results in Section 6, the related work in Section 7, and our conclusions in Section 8.

## 2. PROBLEM FORMULATION

In this section, we formally define the problem of semantic pattern annotation (SPA).

Let $D = \{t_1, t_2, \ldots, t_n\}$ be a database containing a set of transactions $t_i$, which can be itemsets, sequences, or graphs. Let $p_\alpha$ be a pattern (e.g., an itemset, subsequence, or subgraph) in $D$ and $P_D = \{p_1, \ldots, p_l\}$ be the set of all such patterns. We denote the set of transactions in which $p_\alpha$ appears as $D_\alpha = \{t_i | p_\alpha \in t_i, t_i \in D\}$.

*Definition* 1 (*Frequent Pattern*). A pattern $p_\alpha$ is *frequent* in a dataset $D$ if $\frac{|D_\alpha|}{|D|} \geq \sigma$, where $\sigma$ is a user-specified threshold and $\frac{|D_\alpha|}{|D|}$ is called the *support* of $p_\alpha$, usually denoted as $s(\alpha)$.

*Definition* 2 (*Context Unit*). Given a dataset $D$ and the set of frequent patterns $P_D$, a *context unit* is a basic object in $D$ which may carry semantic information and which cooccurs with at least one $p_\alpha \in P_D$ in at least one transaction $t_i \in D$. The set of all such context units satisfying this definition is denoted as $U_D$.

With this general definition, a context unit can be an item, pattern, or even transaction, depending on the specific task and data.

*Definition* 3 (Pattern Context). Given a dataset $D$ and a frequent pattern $p_\alpha \in P_D$, the *context* of $p_\alpha$, denoted as $c(\alpha)$, is represented by a selected set of context units $U_\alpha \subseteq U_D$ such that every $u \in U_\alpha$ cooccurs with $p_\alpha$. Each selected context unit $u$ is also called a *context indicator* of $p_\alpha$, associated with a *strength weight* $w(u, \alpha)$, which measures how well it indicates the semantics of $p_\alpha$.

The following is an example of the context for an itemset pattern in a small dataset with only two transactions. The possible context units for this dataset are single items, itemsets, and transactions, and the context indicators of the itemset pattern are selected from the context units appearing with it in the same transactions.

*Example* 3 (*Pattern Context*).

**Transactions:**

$t_1$ = {diaper, milk, baby carriage, baby lotion, ...}
$t_2$ = {digital camera, memory disk, printer, ...}
...

**Context Units:**
1) items: diaper, milk, printer, ...
2) patterns: {diaper, baby lotion}, ...
3) transactions: $t_1, t_2, \ldots$
**An exemplary frequent pattern** ($\sigma = 0.5$)
$p$ = {diaper, milk}
**Context indicators of** $p$**:**
diaper, baby carriage, {milk, baby lotion}, $t_1, \ldots$

With the preceding definitions, we now define the concept of semantic annotation for a frequent pattern and the related three subproblems.

*Definition* 4 (*Semantic Annotation*). Let $p_\alpha$ be a frequent pattern in a dataset $D$, $U_\alpha$ be the set of context indicators of $p_\alpha$, and $P$ be a set of patterns in $D$. A *semantic annotation* of $p_\alpha$ consists of: (1) a set of context indicators of $p_\alpha$, $I_\alpha \subseteq U_\alpha$, such that $\forall u \in I_\alpha$ and $\forall u' \in U_\alpha - I_\alpha$, $w(u', \alpha) \leq w(u, \alpha)$; (2) a set of transactions $T_\alpha \subseteq D_\alpha$, such that $\forall t \in T_\alpha$ and $\forall t' \in D_\alpha - T_\alpha$, $t$ is more similar to $c(\alpha)$ than $t'$ under some similarity measure; and (3) a set of patterns $P' \subseteq P$ such that $\forall p \in P'$ and $\forall p' \in P - P'$, $c(p)$ is closer to $c(\alpha)$ than $c(p')$.

*Definition* 5 (*Context Modeling*). Given a dataset $D$ and a set of possible context units $U$, the problem of *context modeling* is to select a subset of context units $U$, define a strength measure $w(\cdot, \alpha)$ for context indicators, and construct a model of $c(\alpha)$ for each given pattern $p_\alpha$.

*Definition* 6 (*Transaction Extraction*). Given a dataset $D$, the problem of *transaction extraction* is to define a similarity measure $sim(\cdot, c(\cdot))$ between a transaction and a pattern context, and to extract a set of $k$ transactions $T_\alpha \subseteq D_\alpha$ for frequent pattern $p_\alpha$, such that $\forall t \in T_\alpha$ and $\forall t' \in D_\alpha - T_\alpha$, $sim(t', c(\alpha)) \leq sim(t, c(\alpha))$.

*Definition* 7 (*Semantically Similar Pattern (SSP) Extraction*). Given a dataset $D$ and a set of candidate patterns $P_c$, the problem of *semantically similar pattern (SSP) extraction* is to define a similarity measure $sim(c(\cdot), c(\cdot))$ between the contexts of two patterns, and to extract a set of $k$ patterns $P' \subseteq P_c$ for any frequent pattern $p_\alpha$, such that $\forall p \in P'$ and $\forall p' \in P_c - P'$, $sim(c(p'), c(\alpha)) \leq sim(c(p), c(\alpha))$, where $c(\alpha)$ is the context of $p_\alpha$.

With the aforesaid definitions, we may define the basic task of *semantic pattern annotation (SPA)* as to

(1) select context units and design a strength weight for each unit to model the contexts of frequent patterns;
(2) design similarity measures for the contexts of two patterns, as well as for a transaction and a pattern context; and
(3) for a given frequent pattern, extract the most significant context indicators, representative transactions, and semantically similar patterns to construct a structured annotation.

To refine the basic SPA, we may further extract the supporting transactions for each extracted context indicator to provide further context for a user to understand the pattern. For a similar purpose, we may also extract supporting context indicators for each extracted semantically similar pattern to help a user understand why they are similar.

This problem is challenging in various aspects. First, we do not have prior knowledge on how to model the context of a pattern or select context units when the complete set of possible context units is huge. Second, it is not immediately clear how to analyze pattern semantics, thus the design of the strength weighting function and similarity measure are nontrivial. Finally, since no training data is available, the annotation must be generated in a completely unsupervised way. These challenges, however, also indicate a great advantage of the SPA techniques we will propose, namely that they do not depend on any domain knowledge about the dataset or patterns.

In the following two sections, we present our approaches for modeling the context of a frequent pattern and annotating patterns through semantic context analysis.

## 3. CONTEXT MODELING OF FREQUENT PATTERNS

In this section, we discuss how to model the context of frequent patterns through selecting appropriate context units and defining appropriate strength weights. Given a dataset $D$ and a set of frequent patterns $P_D$, our goal is to select a set of context units which carry semantic information and can discriminate the meanings of the frequent patterns. The discriminating power of each context unit will be captured by its strength weights.

### 3.1 Context Modeling with the Vector-Space Model

The vector-space model (VSM) [Salton et al. 1975] is commonly used in natural language processing and information retrieval to model the content of a text. For example, in information retrieval, a document and a query are both represented as term vectors where each term is a basic concept (e.g., word, phrase), and each element of the vector corresponds to a term weight reflecting the importance of the term. The similarity between documents and queries can thus be measured by the distance between the two vectors in the vector space. For the purpose of semantic modeling, we represent a transaction and the context of a frequent pattern both as vectors of context units. We select VSM because it makes no assumption on vector dimensions and gives the most flexibility to selection of dimensions and weights. Formally, the context of a frequent pattern is modeled as follows.

*Context Modeling.* Given a dataset $D$ and a selected set of context units $U = \{u_1, \ldots, u_m\}$, we represent the context $c(\alpha)$ of a frequent pattern $p_\alpha$ as a vector $\langle w_1, w_2, \ldots, w_m \rangle$, where $w_i = w(u_i, \alpha)$ and $w(\cdot, \alpha)$ is a weighting function. A transaction $t$ is represented as a vector $\langle v_1, v_2, \ldots, v_m \rangle$, where $v_i = 1$ iff $u_i \in t$, otherwise $v_i = 0$.

There are two key issues in a VSM, which are: (1) to select the vector dimensions, and (2) to assign weights for each dimension [Salton et al. 1975].

The choices for both can be flexible. Due to the generality of VSM, the proposed vector-space pattern context model is also quite general and covers different strategies for context unit selection and weighing functions. Naturally, in any specification, the effectiveness of context modeling would be highly dependent on how to select context units and design the strength weights. In the following subsections, we first discuss the generality of the context model, and then discuss specific solutions for the two issues, respectively.

### 3.2 The Generality of Context Modeling and Special Cases

Since the vector-space model is general, there are many special cases of pattern context modeling, corresponding to different strategies to select and weigh the dimensions. Some existing work has explored nonmorphological information of frequent patterns with some concepts related to the "pattern context" defined previously. We now show that the notion of "pattern context" is more general and can cover those concepts as special cases.

In Yan et al. [2005], the authors introduced the profile of an itemset for summarizing itemset patterns, which is represented as a Bernoulli distribution vector. Their goal is to select the most representative patterns to summarize a set of frequent patterns according to their profiles. In fact, this "profile" of a frequent itemset $\alpha$ can be written as a vector $\langle w_p(o_1, \alpha), w_p(o_2, \alpha), \ldots, w_p(o_d, \alpha)\rangle$ over all the single items $\{o_i\}$ in $D$. Here $w_p(o_i, \alpha) = \frac{\sum_{t_j \in D_\alpha} t_j^i}{|D_\alpha|}$, where $t_j^i = 1$ if $o_i \in t_j$, and 0 otherwise. This shows that this profile is actually a special instance of the context model as we have defined it, where single items are selected as context units, and $w_p(\cdot, \alpha)$ is defined as the weighting function for each dimension.

Xin and others proposed a distance measure for mining compressed frequent-pattern sets, which is computed based on the transaction coverage of two patterns [Xin et al. 2005]. Interestingly, this "transaction coverage" is also a specific instance of pattern context. Given a frequent pattern $p_\alpha$, the transaction coverage of $p_\alpha$ can be written as a vector $\langle w_c(t_1, \alpha), w_c(t_2, \alpha), \ldots, w_c(t_k, \alpha)\rangle$ over all the transactions $\{t_i\}$ in $D$, where $w_c(t_i, \alpha) = 1$ if $p_\alpha \in t_i$, and 0 otherwise. We can easily see that in the transaction coverage, transactions are selected as context units and $w_c(\cdot, \alpha)$ is defined as the weight of each dimension.

Pantel and Lin proposed an algorithm to discover different senses of words from text database based on the context of words [Pantel and Lin 2002]. In their approach, the context of each word is represented with a vector, where each dimension is a "context feature" which is similar to a frequent sequential pattern in the text containing the target word. The weighting function of each dimension is then defined as the discounted pointwise mutual information of the word $w$ and the corresponding context feature $c$. It is clear that this vector is also a special case of our context model, where the target frequent patterns are restricted to *words* in the text database.

Besides the aforementioned strategies, there could be many other special cases of general pattern context modeling. In general, the context units can be an arbitrary set of single items, transactions, and more generally, frequent patterns. For example, if each context unit is a set of transactions with the

same timestamp, the context vector will become a time-series vector of the target pattern. The weighting function of each dimension can be binary weighting, Bernoulli distribution, occurrence, or more complicated weighting functions like Jaccard similarity [Jaccard 1908] and mutual information [Church and Hanks 1990], or domain-specific weighting functions such as edit distance [Wagner and Fischer 1974] for sequence patterns and TF-IDF weighting for text patterns [Fang et al. 2004].

It is interesting to notice that the user can also incorporate various constraints on both the selection of context units and the weighting of dimensions. For example, one can restrict the context units to closed sequential patterns as opposed to all frequent patterns, or define each dimension of the VSM to represent a cluster of transactions. Similarly, the user could also incorporate a constraint on the weighting function $w(\cdot, \alpha)$. For example, if we impose the constraint that $\sum_{u_i \in U} w(u_i, \alpha) = 1$, the dimension weights of the context vector of a frequent pattern can be viewed as a probabilistic distribution. This constraint could potentially bridge the gap between our model and many existing information retrieval techniques, where such a probabilistic distribution is commonly used to represent the context of a word or document [Croft and Lafferty 2003].

Covering the concepts in existing work as specific instances, the pattern context model we propose is general and has quite a few benefits. First, it does not assume pattern types. The pattern profile proposed in Yan et al. [2005] assumes that both transactions and patterns are itemsets, thus does not work for other patterns such as sequential and graph patterns. Second, pattern context modeling allows different granularity of context units and different weighting strategies. In many cases, single items are not informative in terms of carrying semantic information (e.g., single nucleotides in DNA sequences), and the semantic information carried by a full transaction is too complex and noisy (e.g., a text document). The context modeling we introduce bridges this gap by allowing various granularities of semantic units, and allows the user to explore pattern semantics at the level that corresponds to her beliefs. Furthermore, this model is adaptive to different strength weighting strategies for context units, where the user's prior knowledge about the dataset and patterns can be easily plugged in. A detailed discussion of how a user's prior knowledge can be incorporated can be found in Section 5. In the rest of Section 3, we will discuss how we select the context units and weight each dimension in a practical way.

### 3.3 Context Unit Selection

With the general definition presented in Section 2, the selection of context units is quite flexible. In principle, any object in the database that carries semantic information or serves to discriminate patterns semantically can be a context unit, thus context units can be single items, transactions, patterns, or any group of items/patterns, depending on the characteristics of the task and data.

Without losing generality, in our work we assume that a *pattern* is the number of minimal units which carry semantic information in a dataset, and thus we select the context units as patterns. All kinds of units can be considered as patterns with a specific granularity. For example, in a sequence database, every item can be viewed as a sequential pattern of length 1, and every transaction as

a sequential pattern which is identical to the transactional sequence. Choosing patterns as context units is task dependent, and can usually be optimized with prior knowledge about the task and data. For example, we can use words as context units in a text database, while in a graph database we prefer subgraph patterns to be context units, since single items (i.e., vertices and edges) are noninformative.

This general strategy gives much freedom to select context units. However, selecting patterns of various granularities may cause redundancy of context because these patterns are highly redundant. As discussed in previous sections, we expect the context units not only to carry semantic information, but also to be as discriminating as possible to indicate the meanings of a pattern. However, when various granularities of patterns are selected as context units, some units will become less discriminative, and, more severely, some will become redundant. For example, when the pattern "mining subgraph" is added as a context unit, the discriminative power of other units like "mining frequent subgraph" and "subgraph" will be weakened. This is because the transactions containing the pattern "mining subgraph" always contain "subgraph", and likely also contain "mining frequent subgraph", which means that these patterns are highly dependent and not sufficiently discriminative to indicate the semantics of the frequent patterns cooccurring with them. This redundancy also brings a lot of unnecessary dimensions into the context vector-space where the dimensionality is already very high. This redundancy in dimensions will affect both the efficiency and accuracy of distance computation between two vectors, which is essential for SPA. In our work, we examine different techniques to remove the redundancy of context units without losing their semantic discriminating power.

3.3.1 *Redundancy Removal: Existing Techniques.* One may first think of using existing techniques such as pattern summarization and dimension reduction to remove the redundancy of context units.

While the context units can be any pattern in principle, we are practically uninterested in those with very low frequency in the databases. Therefore, the context units we initially include are *frequent patterns*. There exist methods for summarizing frequent patterns with $k$ representative patterns [Yan et al. 2005], but they only work for itemset patterns and are not general enough for our purpose.

Some techniques, such as LSI [Deerwester et al. 1990], have been developed to reduce the dimensionality in high-dimensional spaces, especially for text data. However, these techniques aim to mitigate the sparseness of data vectors by reducing the dimensionality, and are not tuned for removing the "redundant" dimensions. This is because all these dimensionality reduction techniques consider that each dimension is "important" and the information it carries will always be preserved or propagated into the new space. This is, however, different from our goal of redundancy removal. For example, let $d_1$ and $d_2$ correspond to the patterns "AB" and "ABC". If we consider $d_2$ to be redundant with respect to $d_1$, we do not expect the information of $d_2$ to be preserved after removal of $d_2$.

3.3.2 *Redundancy Removal: Closed Frequent Patterns.* Since neither the pattern summarization nor the dimensionality reduction technique is directly

applicable to our problem, we examine alternative strategies. Noticing that the redundancy of context units is likely to be caused by the inclusion of both a frequent pattern and its subpatterns, we explore closed frequent [Pasquier et al. 1999] and maximal frequent patterns [Bayardo 1998] to solve this problem.

A maximal frequent pattern is a frequent pattern which does not have a frequent superpattern. It is easy to show that a maximal frequent pattern is not appropriate for this problem since it may lose important discriminative units. For example, the frequent pattern "data cube", although not maximal, indicates different semantics from the frequent pattern "prediction data cube" and thus should not be removed.

*Definition* 8 (*Closed Frequent Pattern*). A frequent pattern $p_\alpha$ is closed if and only if there exists no superpattern $p_\beta$ of $p_\alpha$, such that $D_\alpha = D_\beta$.

We assume that a context unit is not redundant only if it is a closed pattern. This assumption is reasonable because $\forall p_\alpha \in P_D$, if $p_\alpha$ is not closed, there is always another frequent pattern $p_\beta \in P_D$, where $p_\alpha \subseteq p_\beta$ and $\forall t_i \in D$, we have $p_\alpha \in t_i \Leftrightarrow p_\beta \in t_i$. This indicates that we can use $p_\beta$ as a representative of $p_\alpha$ and $p_\beta$ without losing any semantic discriminating power. Therefore, in our work we use closed frequent patterns as our initial set of context units. The algorithms for mining different kinds of closed frequent patterns can be found in Pasquier et al. [1999] and Yan et al. [2003].

3.3.3 *Redundancy Removal: Microclustering.* However, as stated in Yan et al. [2005], a small disturbance within the transactions may result in hundreds of subpatterns that could have different supports which cannot be pruned by closed frequent pattern mining. These subpatterns are usually with supports only slightly different from that of the master pattern. Therefore, their discriminating power for the semantics of the frequent patterns is very weak when their master patterns are also included as a context unit. We present clustering methods to further remove redundancy from closed frequent patterns.

*Microclustering* is usually employed as a preprocessing step to group data points from, presumably, the same cluster to reduce the number of data points. In our work, we first introduce a distance measure between two frequent patterns and then introduce two microclustering algorithms to further group the closed frequent patterns.

*Definition* 9 (*Jaccard Distance*). Let $p_\alpha$ and $p_\beta$ be two frequent patterns. The Jaccard distance between $p_\alpha$ and $p_\beta$ is computed as

$$D(p_\alpha, p_\beta) = 1 - \frac{|D_\alpha \cap D_\beta|}{|D_\alpha \cup D_\beta|}.$$

Jaccard distance [Jaccard 1908] is commonly applied to cluster data based on its cooccurrence in transactions. Our need is to group patterns that tend to appear in the same transactions, which is well captured by Jaccard distance. Jaccard distance has also been applied to pattern clustering in Xin et al. [2005].

With Jaccard distance, we expect to extract clusters such that the distances between inner-cluster units are bounded. We present two microclustering algorithms as follows.

**Algorithm 1.** Hierarchical Microclustering

Input: Transaction dataset D,
A set of $n$ closed frequent patterns, $\mathcal{P} = \{p_1, \ldots, p_n\}$
Threshold of distance, $\gamma$
Output: A set of patterns, $\mathcal{P}' = \{p'_1, \ldots, p'_k\}$

1: initialize $n$ clusters $C_i$, each as a closed frequent pattern;
2: compute the Jaccard Distance $d_{ij}$ among $\{p_1, \ldots, p_n\}$;
3: set the current minimal distance $d = min(d_{ij})$;
4: **while** $(d < \gamma)$
5: select $d_{st}$ where $(s, t)$ = argmin$_{i,j} d_{ij}$;
6: merge clusters $C_s$ and $C_t$ into a new cluster $C_u$;
7: **foreach** $C_v \neq C_u$
8: compute $d_{uv} = max(d_{\alpha\beta})$ where $p_\alpha \in C_u$, $p_\beta \in C_v$;
9: **foreach** $C_u$;
10: **foreach** $p_\alpha \in C_u$;
11: compute $\bar{d}_\alpha = avg(d_{\alpha\beta})$ where $p_\beta \in C_u$;
12: add $p_\alpha$ into $\mathcal{P}'$, where $\alpha$ = argmin$_i(\bar{d}_i)$;
13: **return**

**Algorithm 2.** One-Pass Microclustering

Input: Transaction dataset D,
A set of $n$ closed frequent patterns, $\mathcal{P} = \{p_1, \ldots, p_n\}$
Threshold of distance, $\gamma$
Output: A set of patterns, $\mathcal{P}' = \{p'_1, \ldots, p'_k\}$

1: initialize 0 clusters;
2: compute the Jaccard Distance $d_{ij}$ among $\{p_1, \ldots, p_n\}$;
3: **foreach** $(p_\alpha \in \mathcal{P})$
4: **foreach** cluster $C_u$
5: $\tilde{d}_{\alpha,u} = max(d_{\alpha\beta})$ where $p_\beta \in C_u$;
6: $v$ = argmin$_u(\tilde{d}_{\alpha,u})$;
7: **if**$(\tilde{d}_{\alpha,v} < \gamma)$
8: assign $p_\alpha$ to $C_v$
9: **else**
10: initialize a new cluster $C = \{p_\alpha\}$
11: **foreach** $C_u$;
12: **foreach** $p_\alpha \in C_u$;
13: compute $\bar{d}_\alpha = avg(d_{\alpha\beta})$ where $p_\beta \in C_u$;
14: add $p_\alpha$ into $\mathcal{P}'$, where $\alpha$ = argmin$_i(\bar{d}_i)$;
15: **return**

In the hierarchical microclustering method presented as Algorithm 1, we iteratively group two clusters of patterns with the smallest distance, where the distance between two clusters is defined as the Jaccard distance between the farthest patterns in the two clusters. The algorithm terminates when the

minimal distance between clusters becomes larger than a user-specified threshold $\gamma$. The second algorithm, which we call one-pass microclustering, iteratively assigns a closed frequent pattern $p_\alpha$ to its nearest cluster if the distance is below $\gamma$, where the distance between $p_\alpha$ and a cluster $C$ is defined as the Jaccard distance between $p_\alpha$ and its farthest pattern in $C$. Both algorithms give us a set of microclusters of closed frequent patterns. They both guarantee that the distance between any pair of patterns in the same cluster is below $\gamma$. Only the medoid of each cluster is selected as a context unit. By varying $\gamma$, a user can select context units with various levels of discriminating power of pattern semantics. It is clear that Algorithm 2 only passes the pattern set once and thus is more efficient than the hierarchical algorithm, at the expense that the quality of clusters depends on the order of patterns. The performance of these two methods is compared in Section 6.

### 3.4 Strength Weighting for Context Units

Once the context units are selected, the remaining task is to assign a weight to each dimension of the context model that represents how strongly the context unit corresponding to this dimension indicates the meaning of a given pattern. If the user has prior knowledge about which context units can better indicate the meaning of the given pattern, the user can incorporate this knowledge to the strength weighting function and assign the corresponding dimensions higher weights (see Section 5). In this section, we assume that no such prior knowledge is available.

Intuitively, the strongest context indicators for a pattern $p_\alpha$ should be those units that frequently cooccur with $p_\alpha$ but infrequently cooccur with others. In practice, many types of weighting functions can be used to measure the strength of a context indicator. For example, we can let the weight of a context indicator $u$ for $p_\alpha$ be the number of transactions with both $u$ and $p_\alpha$. In order to choose a reasonable weighting function, we next define several constraints that a reasonable weighting function should satisfy.

Given a set of context indicators $U$ and a frequent pattern $p_\alpha$, a strength weighting function $w(\cdot, p_\alpha)$ is reasonable if $\forall u_i \in U$:

(1) $w(u_i, p_\alpha) \leq w(p_\alpha, p_\alpha)$; the best semantic indicator of $p_\alpha$ is itself.
(2) $w(u_i, p_\alpha) = w(p_\alpha, u_i)$; two patterns are equally strong to indicate the meanings of each other.
(3) $w(u_i, p_\alpha) = 0$ if the appearance of $u_i$ and $p_\alpha$ is independent; $u_i$ cannot indicate the semantics of $p_\alpha$.

An obvious choice is cooccurrences, which, however, may not be a good measure. On the one hand, it does not satisfy constraint 3. On the other hand, we want to penalize the context units that are globally common patterns in the collection. This means that although they may cooccur many times with $p_\alpha$, they may still not be a good context indicator for $p_\alpha$ because they also cooccur frequently with others. In general, context units that are strongly correlated to $p_\alpha$ should be weighted higher. In our work, we introduce a more principled measure.

Mutual information (MI) [Cover and Thomas 1991] is widely used to measure the mutual independency of two random variables in information theory, which intuitively measures how much information one random variable tells about the other. The definition of mutual information is given as follows.

*Definition* 10 (*Mutual Information*). Given two frequent patterns $p_\alpha$ and $p_\beta$, let $X = \{0, 1\}$ and $Y = \{0, 1\}$ be two random variables for the appearance of $p_\alpha$ and $p_\beta$, respectively. *Mutual information* $\mathrm{I}(X; Y)$ is computed as

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x, y) \log \frac{P(x, y)}{P(x)P(y)},$$

where $P(x = 1, y = 1) = \frac{|D_\alpha \cap D_\beta|}{|D|}$, $P(x = 0, y = 1) = \frac{|D_\beta| - |D_\alpha \cap D_\beta|}{|D|}$, $P(x = 1, y = 0) = \frac{|D_\alpha| - |D_\alpha \cap D_\beta|}{|D|}$, and $P(x = 0, y = 0) = \frac{|D| - |D_\alpha \cup D_\beta|}{|D|}$. In our experiments, we use standard Laplace smoothing to avoid zero probability.

It can be easily proved that mutual information satisfies all three constraints given earlier and favors the strongly correlated units. In our work, we use mutual information to model the indicative strength of the context units selected.

Given a set of patterns as candidate context units, we apply closeness testing and microclustering to remove redundant units from this initial set. We then use mutual information as the weighting function for each indicator selected. Given a frequent pattern, we apply semantic analysis with its context model and generate annotations for this pattern, as discussed in the following section.

## 4. SEMANTIC ANALYSIS AND PATTERN ANNOTATION

Let $U = \{u_1, u_2, \ldots, u_k\}$ be a selected set of $k$ context units and $w(\cdot, p_\alpha)$ be a unit weighting function defined on any frequent pattern $p_\alpha \in I(\cdot; p_\alpha)$. The context model or context vector $c(\alpha)$ for $p_\alpha$ is $\langle w(u_1, p_\alpha), w(u_2, p_\alpha), \ldots, w(u_k, p_\alpha)\rangle$.

As introduced in Section 1, we make the assumption that frequent patterns are semantically similar if their contexts are similar. In our work, we analyze the semantics of frequent patterns by comparing their context models. Formally, we have the next definition.

*Definition* 11 (*Semantical Similarity*). Let $p_\alpha, p_\beta, p_\delta$ be three frequent patterns in $P$ and $c(\alpha), c(\beta), c(\delta) \in V_k$ be their context models. Let $sim(c(\cdot), c(\cdot)) : V_k \times V_k \longrightarrow \mathbb{R}^+$ be a similarity function of two context vectors. If $sim(c(\alpha), c(\beta)) > sim(c(\alpha), c(\delta))$, then we say that $p_\alpha$ is semantically more similar to $p_\beta$ than to $p_\delta$ by $sim(c(\cdot), c(\cdot))$.

Because of the generality of context modeling, the choice of semantical similarity measure is also very flexible. In practice, the selection of semantical similarity measure should be consistent with how the context vectors are constructed. In Yan et al. [2005], the authors compare the profiles of two patterns with the Kullback-Leibler divergence, which is a commonly used distance measure to compare two probabilistic distributions. Xin et al. [2005] use the Jaccard distance to measure the similarity of two binary vectors. In our framework, we can support any general similarity measure defined on the context vectors.

*Cosine* is widely used to compute the similarity between two vectors, and is well explored in information retrieval to measure the relevance between a document and a query if both are represented with a vector-space model [Salton et al. 1975]. Thus, as a specific choice, we may use cosine similarity of two context vectors to measure the semantic similarity of two corresponding frequent patterns; other potential measures include the *Pearson coefficient*, *Euclidean distance*, etc.

Formally, the cosine similarity of two context vectors is computed as

$$sim(c(\alpha), c(\beta)) = \frac{\sum_{i=1}^{k} a_i * b_i}{\sqrt{\sum_{i=1}^{k} a_i^2} * \sqrt{\sum_{i=1}^{k} b_i^2}},$$

where $c(\alpha) = \langle a_1, a_2, \ldots, a_k \rangle$ and $c(\beta) = \langle b_1, b_2, \ldots, b_k \rangle$.

With the context model and semantical similarity measure, we now discuss how to generate semantic annotations for frequent patterns.

### 4.1 Extracting Strongest Context Indicators

Let $p_\alpha$ be a frequent pattern and $c(\alpha)$ be its context model, which is defined in this work as a context vector $\langle w_1, w_2, \ldots, w_k \rangle$ over a set of context units $U = \{u_1, u_2, \ldots, u_k\}$. As defined in Section 2, $w_i$ is a weight for $u_i$ which tells how well $u_i$ indicates the semantics of $p_\alpha$. Therefore, the goal of extracting strongest context indicators is to extract a subset of $k'$ context units $U_\alpha \subseteq U$ such that $\forall u_i \in U_\alpha$ and $\forall u_j \in U - U_\alpha$, we have $w_i \geq w_j$.

With a strength weighting function $w(\cdot, p_\alpha)$, for example, mutual information as introduced in Section 3, we compute $w_i = w(u_i, p_\alpha)$, rank $u_i \in U$ with $w_i$ in descending order, and select the top $k'$ $u_i$'s.

To help a user interpret the extracted context indicators, we can further extract the major supporting transactions for each context indicator. Specifically, given an extracted context indicator $u_i$, we can easily obtain all the transactions in which the pattern to be annotated ($p_\alpha$) and $u_i$ cooccur. Such a set of transactions (denoted by $S(u_i)$) can be regarded as the supporting transactions for $u_i$. We can then take the following centroid transaction as a major supporting transaction for the context indicator $u_i$.

$$supp(u_i) = \operatorname*{argmax}_{t \in S(u_i)} \sum_{t' \in S(u_i), t' \neq t} sim(t, t')$$

If needed, we can also provide more than one supporting transaction using the previous formula.

### 4.2 Extracting Representative Transactions

Let $p_\alpha$ be a frequent pattern, $c(\alpha)$ be its context model, and $D = \{t_1, \ldots, t_l\}$ be a set of transactions, our goal is to select $k_t$ transactions $T_\alpha \subseteq D$ with a similarity function $s(\cdot, p_\alpha)$, such that $\forall t \in T_\alpha$ and $\forall t' \in D - T_\alpha$, $s(t, p_\alpha) \geq s(t', p_\alpha)$.

To achieve this, we first represent a transaction as a vector in the same vector space as the context model of the frequent pattern $p_\alpha$, that is, over $\{u_1, u_2, \ldots, u_k\}$. Then, we use the cosine similarity presented in Section 3 to compute the similarity between a transaction $t$ and the context of $p_\alpha$. The rest

is again a ranking problem. Formally, let $c(t) = \langle w'_1, w'_2, \ldots, w'_k \rangle$, where $w'_i = 1$ if $u_i \in t$ and $w'_i = 0$ otherwise. We compute $sim(c(t), c(\alpha))$ for each $t \in T_\alpha$, rank them in descending order, and select the top $k_t$ $t$'s.

### 4.3 Extracting Semantically Similar Patterns

Let $p_\alpha$ be a frequent pattern, $c(\alpha)$ be its context model, and $P_c = \{p_1, \ldots, p_c\}$ be a set of frequent patterns believed to be good candidates for annotating the semantics of $p_\alpha$, that is, as synonyms, thesauri, or more generally as SSPs. Our goal is to extract a subset of $k_c$ patterns $P'_c \subseteq P_c$ whose contexts are most similar to $p_\alpha$. Formally, let $\{c(p_1), \ldots, c(p_c)\}$ be the context vectors for $\{p_1, \ldots, p_c\}$. We compute $sim(c(p_i), c(\alpha))$ for each $p_i \in P_c$, rank them in descending order, and select the top $k_c$ $p_i$'s.

Note that the candidate SSP set for annotation is quite flexible. It can be the whole set of frequent patterns in $D$, or a user-specified set of patterns based on his prior knowledge. It can be a set of homogenous patterns with $p_\alpha$, or a set of heterogenous patterns. For example, it can be a set of patterns or terminology from the domain with which a user is familiar, and is used to annotate patterns from an unfamiliar domain. This brings great flexibility to apply general SPA techniques to different tasks. By exploring different types of candidate SSPs, we can find quite a few interesting applications of semantic pattern annotation, which are discussed in Section 6.

In particular, those extracted semantically similar patterns of the "same type" as pattern $p_\alpha$ are especially interesting. We define the type of a pattern as the set of types of all the elements of the pattern. Formally, let $\tau(x)$ be the type of the data value $x$. The type of pattern $p$ (denoted by $\tau(p)$) can be defined as

$$\tau(p) = \bigcup_{x \in p} \tau(x),$$

where $x \in p$ is true if and if only if the value $x$ is contained in pattern $p$. For example, if $p_\alpha$ is a frequent itemset where each item is an author in a bibliographic database, then any pattern with a set of authors would be regarded as having the same type as $p_\alpha$. Similarly, if $p_\alpha$ is a cooccurring pattern of an author and words in the title, then any other author-word cooccurring patterns would be of the same type as $p_\alpha$. By restricting the semantically similar patterns to those with the same type as $p_\alpha$, we can extract patterns whose similarity to $p_\alpha$ can be easily interpreted.

Furthermore, for any given SSP extracted, we can help a user to understand in what sense it is similar to $p_\alpha$ by further extracting the context indicators that contribute most to the similarity. The contribution of a context indicator can be measured by the change in similarity when we take it out. Formally, let $p_i$ be an SSP extracted for $p_\alpha$ and $x \in I_{p_\alpha}$ be a context indicator. Let $c(p_i)$ and $c(p_\alpha)$ be the context vectors for $p_i$ and $p_\alpha$, respectively. Let $c_{-x}(p_i)$ and $c_{-x}(p_\alpha)$ be the context vectors formed by *excluding* indicator $x$ for $p_i$ and $p_\alpha$, respectively. The contribution of $x$ to the similarity of $p_i$ and $p_\alpha$ can be measured by

$$\delta(x, p_i, p_\alpha) = sim(c(p_i), c(p_\alpha)) - sim(c_{-x}(p_i), c_{-x}(p_\alpha)).$$

We can then choose the indicators with the highest $\delta(x, p_i, p_\alpha)$ as supporting indicators for SSP $p_i$. When we use the cosine similarity measure, we may also simply measure the contribution of $x$ by $w(x, p_i)w(x, p_\alpha)$.

## 5. SEMANTIC ANNOTATION WITH PRIOR KNOWLEDGE FROM USERS

In Sections 3 and 4, we discussed how the context vectors and semantical similarity measures of frequent patterns can be constructed and selected in a general way. In the proposed approaches, we assume that no prior knowledge is available, which makes our approaches generally applicable to any application.

However, in any specific application, a user often has some prior knowledge which can provide useful guidance for context modeling and semantic annotation of a given pattern, thus leading to potentially much more useful annotations. In this section, we show that our context analysis framework could easily incorporate different prior knowledge.

Let us begin with the following example. We shall suppose that a data mining researcher who has little knowledge about biology is using the pattern annotation system to understand some frequent patterns in bioinformatics literature. In order to understand a frequent itemset of unknown authors, the user will likely look at the strongest context indicators of the pattern, which may include, for example, authors who have collaborated with the authors in the pattern. Intuitively, some context indicators (e.g., computer science authors) will be more meaningful for the user than others (e.g., biology authors). In other words, a useful context indicator should be a computer science author, especially a data mining professor. In general, among all context indicators, well-known authors and those with whom the user is most familiar are most semantically meaningful to the user.

To help the system generate such useful context indicators, we may incorporate the user's prior knowledge by restricting the context indicators to those authors known to the user (e.g., data mining authors) and assign higher weights to the authors whose research the user is familiar with. Similarly, when annotating the frequent pattern "gene pathway", "gene network" is a much better semantically similar pattern than "regulatory pathway" for a data mining researcher, as the former is easier to understand.

Another problem of a frequent pattern is that its semantic meaning may be ambiguous. Indeed, it is common that a frequent pattern may be interpreted differently in different contexts or by different users. For example, the frequent itemset {michael, jordan} may mean either the legendary basketball player or the machine learning professor in UC Berkeley. When annotating such a pattern without any prior knowledge, the context units, representative transactions, and semantically similar patterns will all likely be a mixture of these two meanings. This is not desirable for a particular user who, presumably, would favor one of the two meanings. However, if the system knows some information about the "preference" of the user in advance (e.g., the domain that the user is interested in), it could disambiguate the annotations of the target pattern by boosting the those within the favored domain and weakening those in other domains. In the previous example, a computer scientist may expect

to see that the highest ranked context indicators, transactions, and SSPs for the pattern {michael, jordan} are patterns related to computer science, such as "topic models" and {david, blei}.

Formally, in our context modeling and semantic annotation framework, a user's prior knowledge can be incorporated in the following ways.

(1) *Guiding Context Unit Selection.* Let $U_p = \{u_{p1}, \ldots, u_{pl}\}$ be a set of frequent patterns which are useful to indicate the context of a given pattern $p_\alpha$ according to the user's prior knowledge. The context unit set $U$ can be selected such that $U = U_p$ or $U_p \subset U$. This means the user can either define the dimensions of the context model, or provide some seeds of the context units. For example, $u_{pi}$ can be a terminology, entity, phrase, or other specific patterns in the domain with which that the user is familiar. In practice, we can also incorporate a pattern-independent weighting function of the context units, $w_p(\cdot)$, and use a threshold $\delta$ to prune the context unit $u_j$ when $w_p(u_j) < \delta$.

(2) *Adjusting Context Unit Weight.* Let $U$ be a set of context units, and $w_p(\cdot)$ be a weighting function such that $w_p(u_i)$ measures the importance of context unit $u_i$ according to the user's prior knowledge. Given a general strength weighting function $w(\cdot, \alpha)$ as given in Section 3, a user-specific weighting function for the dimensions of the context model can be defined as $W(u_i, \alpha) = F(w_p(u_i), w(u_i, \alpha))$, where $F$ is a combination function of $w_p$ and $w$. For example,

$$W(u_i, \alpha) = (1 - \lambda) \log \frac{w(u_i, \alpha)}{\sum_{u_j \in U} w(u_j, \alpha)} + \lambda \log \frac{w_p(u_i)}{\sum_{u_j \in U} w_p(u_j)},$$

where $\lambda$ is a parameter to control the influence of the prior. Moreover, $w_p(\cdot)$ could be some importance measures, such as the PageRank [Brin et al. 1997], authority [Kleinberg 1999], statistical significance, or other measures that are independent of the target pattern $p_\alpha$.

(3) *Restricting the Candidate SSP Set.* Let $C_p = \{p_1, \ldots, p_l\}$ be a set of frequent patterns that are semantically meaningful according to the user's prior knowledge. The candidate SSP set $P_c$ can be selected as a set of frequent patterns such that $\forall p \in P_c$, $\exists p' \in C_p$, such that $p' = p$, or $p' \sim p$ ($p'$ is related to $p$). This means that the user could either define a candidate SSP set or provide some seeds which can be naturally extended to a larger candidate SSP set. For example, $C_p$ can be an ontology, a set of entities, famous authors in the same domain of the target pattern, or entities in a different domain/language.

(4) *Computing Context Similarity.* A user's prior knowledge is also useful to guide the selection of a similarity measure for two context vectors. On the one hand, this similarity measure can be selected corresponding to specific characteristics of the database, patterns, and context models. For example, when the context is modeled as a probabilistic distribution (see Section 3.2), KL divergence can be selected accordingly. On the other hand, since tasks 2 and 3 of semantic annotation are defined as ranking problems, some supervised ranking methods (e.g., Ranking SVM [Joachims 2002] and RankNet

[Burges et al. 2005]) can also be applied to learn a ranking function (semantic similarity function) for the example transactions or candidate SSPs, and the user's prior knowledge can be utilized to provide training examples. This is also an interesting future direction in which to extend this work.

(5) *Constraining the Supporting Transactions and Supporting Context Indicators.* A user's prior knowledge can also be used to influence the selection of supporting transactions for an extracted context indicator so that the supporting transactions would be more meaningful. Similarly, it can also be exploited to bias our selection of supporting context indicators for an SSP so that the selected context indicators would be more meaningful for the user.

Although we do not explore all these possibilities for incorporating a user's prior knowledge, it is clear that our general context modeling and semantic annotation framework can easily incorporate a user's prior knowledge in multiple ways. The specific way of using the prior knowledge will inevitably depend on the data, frequent patterns, and type of prior knowledge in a specific application.

## 6. EXPERIMENTS AND RESULTS

In this section, we present experiment results on three different datasets to show the effectiveness of the semantic pattern annotation technique for various real-world tasks.

### 6.1 DBLP Dataset

The first dataset we use is a subset of the DBLP dataset.[2] It contains papers from the proceedings of 12 major conferences in database and data mining. Each transaction consists of two parts: the authors and the title of the corresponding paper. We consider two types of patterns: (1) frequent coauthorship, each pattern of which is a frequent itemset of authors; and (2) frequent title terms, each of which is a frequent sequential pattern of the title words. The goal of experiments on this dataset is to show the effectiveness of the SPA to generate a dictionary-like annotation for frequent patterns, which can help a user understand the meanings of the annotated patterns. Our experiments are designed as follows.

(1) Given a set of authors/coauthors, annotate each of them with their strongest context indicators, the most representative titles from their publications, and the coauthors or title patterns which are most semantically similar. Note that the most representative titles do not necessarily mean their most influential work, but rather the titles which best distinguish their work from others.

(2) Given a set of title terms (sequential patterns), annotate each of them with their strongest context indicators, most representative titles, most similar terms, and most representative author/coauthors. Note again that the most

[2]http://www.informatik.uni-trier.de/~ley/db/

Table I. Effectiveness of Microclustering

| Medoids | Cluster Members |
|---|---|
| mine | data, mine, data mine |
| mine associate rule | rule, associate, associate rule, mine rule mine associate, mine associate rule |
| mine stream | mine data, mine stream, data stream, mine data stream |

representative author/coauthors are not necessarily the most well-known ones, but rather the authors who are most strongly correlated to the topics (terms).

In both experiments, we use the tools FP-Close [Grahne and Zhu 2003] and CloSpan [Yan et al. 2003] to generate closed frequent itemsets of coauthors and closed sequential patterns of title terms, respectively. The title words are stemmed by the Krovetz stemmer [Krovetz 1993], which converts the morphological variations of each English word to root form. We set the minimum support for frequent itemsets as 10 and sequential patterns as 4, which outputs 9926 closed sequential patterns. We use the one-pass microclustering algorithm discussed in Section 3 to remove redundancy from these sequential patterns and get a smaller set of 3443 patterns, with $\gamma = 0.9$ (the average Jaccard distance between these patterns is $>0.95$).

Table I shows the medoids and cluster members of three microclusters generated by the one-pass microclustering algorithm discussed in Section 3, all of which begin with the term "mine". We see that different variations of the same concept are grouped into the same cluster, although all of them are closed patterns. This successfully reduces the pattern redundancy. It is interesting to see that the patterns "data mine" and "mine data" are assigned to different clusters, which cannot be achieved by existing pattern summarization techniques such as Yan et al. [2005]. The results generated by hierarchical microclustering are similar.

In Table II, we selectively show the results of semantic pattern annotations. We see that the SPA system can automatically generate dictionary-like annotations for different kinds of frequent patterns. For frequent itemsets like coauthorship or single authors, the strongest context indicators are usually their other coauthors and discriminative title terms that appear in their work. The semantically similar patterns extracted also reflect the authors and terms related to their work. However, these SSPs may not even cooccur with the given pattern in a paper. For example, the patterns "jianyong_wang", "jiong_yang&philip_s_yu&wei_wang" actually do not cooccur with the pattern "xifeng_yan&jiawei_han", but are extracted because their contexts are similar. For the single-author pattern "christos_faloutsos", describing a database and data mining researcher, the annotation is also quite meaningful. In both cases, the SSPs contain authors with similar research interests.

We also present the annotations generated for title terms which are frequent sequential patterns. Their strongest context indicators are usually the authors who tend to write them in the titles of their papers, or the terms that

Table II. Annotations Generated for Frequent Patterns in DBLP Dataset

| Pattern | Type | Annotations |
|---|---|---|
| xifeng_yan jiawei_han (SSP set = co-author patterns) | I | graph; philip_s_yu; mine close; mine close frequent; index approach; graph pattern; sequential pattern; |
| | T | gspan graph-base substructure pattern mine |
| | T | mine close relational graph connect constraint |
| | T | clospan mine close sequential pattern large database |
| | S | jiawei_han&philip_s_yu; jian_pei&jiawei_han; jianyong_wang; jiong_yang&philip_s_yu& wei_wang; |
| christos_faloutso (SSP set = co-author patterns) | I | spiros_papadimitriou; fast; use fractal; graph; use correlate; |
| | T | multiattribute hash use gray code |
| | T | recovere latent time-sery their observe sum network tomography particle filter |
| | T | index multimedia database tutorial |
| | S | spiros_papadimitriou&christos_faloutso; spiros_papadimitriou; flip_korn; timos_k_selli; ramakrishnan_srikant, ramakrishnan_srikant&rakesh_agrawal |
| information retrieval | I | w_bruce_croft; web information; monika_rauch_henzinger; james_p_callan; full-text; |
| | T | web information retrieval |
| | T | language model information retrieval |
| | S | information use; web information; probabilist information; information filter; text information |
| xquery | I | xquery stream; murali_mani; jens_teubner; tree efficient |
| | T | implement xquery |
| | T | xquery query language xml |
| | S | xquery stream; stream xml; complex language; function query language; estimate xml; |

*Note*: "I" means context indicators; "T" means representative transactions; "S" means semantically similar patterns. We exclude the 12 most frequent and noninformative English words from the collection when extracting frequent patterns.

tend to coappear with them. Their SSPs usually provide interesting concepts or descriptive terms which are close to their meanings, such as "information retrieval → information filter", "xquery → complex language, function query language".

In both scenarios, the representative transactions extracted give us the titles of papers that well capture the meaning of the given patterns. We only show the title words in Table II for each transaction.

Once we extracted the context indicators and SSPs, we can further pull out supporting transactions and supporting context indicators correspondingly, which could give the user more detailed understanding of why the indicators and SSPs are selected. Some results are presented in Table III.

An interesting finding from these results is that the supporting context indicators could be different for two SSPs of the same target pattern (e.g.,

Table III. Supporting Transactions for Context Indicators, and Supporting Context Indicators for Semantically Similar Patterns

| Target Pattern | Context Indicator | Supporting Transactions |
|---|---|---|
| christos_faloutso | spiros_papadimitriou | Streaming pattern discovery in multiple time-series; Adaptive, hands-off stream mining |
| information retrieval | w_bruce_croft | Language models for information retrieval; Fast incremental indexing for full-text information retrieval |
| **Target Pattern** | **SSP** | **Supporting Context Indicators** |
| information retrieval | information filter | information; model; index; structure; latent semantic; collaborative filter; manage information; language |
| xifeng_yan jiawei_han | jian_pei jiawei_han | jiawei_han; mine; mine close; frequent pattern; mine sequential pattern; mine pattern large database |
| xifeng_yan jiawei_han | jiawei_han philip_s_yu | jiawei_han; mine; graph; substructure; graph approach mine close; index approach; similar search |

jian_pei&jiawei_han and jiawei_han&philip_s_yu, for xifeng_yan&jiawei_han). Indeed, in reality, the semantics of a pattern may have multiple aspects, and a semantically similar pattern may be similar to only one of these. The results show that our annotation method is capable of distinguishing such semantical aspects (e.g., "graph indexing/mining" and "frequent pattern mining" for xifeng_yan&jiawei_han).

### 6.2 Incorporating a User's Prior Knowledge

In this section, we test the performance of semantic pattern annotation on incorporating a user's prior knowledge. We use the same dataset as in Section 6.1. In order to evaluate the effectiveness of incorporating a user's prior, we need to select such a user and provide his/her preferences to the system, which is usually difficult. In this experiment, we simulate a user with an author in the DBLP dataset, and simulate the user's prior knowledge with the context model of the author. The basic assumption is that the publication context of an author would help the system to understand the user's information need.

Specifically, we represent an author's prior knowledge with its context vector, and utilize such a vector to guide the selection of context units, adjust the weighting of context units, and to select appropriate candidate SSPs to annotate the target pattern. The original weighting function for a context unit, $w(u_i, \alpha)$, is the simple cooccurrence of $u_i$ and $\alpha$. The results of this experiment are selectively presented in Table IV.

In the first example in Table IV, we simulate two users with two authors from the DBLP dataset, and weight the context indicators of a target pattern from their perspectives, respectively. The third column shows the prior knowledge of a user (i.e., the context indicators of the corresponding author), and the fourth column presents the top context indicators reweighted with the user's prior. When annotating the author pattern "gerhard_weikum" from the view of two other researchers, we see that one "simulated user" ranks the context indicators

Table IV. Reweighted Context Indicators with User's Prior Knowledge

| Target | Simulated User | User Prior (Context) | Context Indicators w/prior |
|---|---|---|---|
| gerhard_weikum | No Prior | N/A | gerhard_weikum; peter_muth; hans-j; guarantee; transact; engine; christof_hasse;† |
| | surajit_chaudhuri | vivek_r_narasayya; index; rajeev_motwani; microsoft serve; | index; automate; database tune; evaluate; database system;† |
| | ihab_f_ilya | walid_g_aref; query optimize; support query database; join; top-k query; | rank; top-k query; database system; engine; guarantee† |
| mine (data, mine, data_mine) | No Prior | N/A | mine; use; efficient; algorithm; mine associate rule; application mine; approach; ‡ |
| | andrew_w_moore | detect; jeremy_kubica; algorithm; detect cluster; tractable; cluster; | classification; cluster; learn; probabilistic data; bayesian network; ‡ |
| | rakesh_agrawal | mine; ramakrishnan_srikant; h_v_jagadish; use; technology; application; | mine associate rule; ramakrishnan _srikant; mine sequential pattern; mine application; efficient; ‡ |

†: selected from the top 10 ranked context indicators, $\lambda = 0.5$.
‡: selected from the top 20 ranked context indicators, $\lambda = 0.5$.

like "database tuning", "automate" highly, and the other ranks the indicators such as "rank", "top-k query" highly. They also both rank "database" highly. This is consistent with reality, since "automatic tuning of database" and "database ranking" are two of the target author's major research interests.

In the other example, we simulated the prior knowledge of two well-known data mining researchers, and try to incorporate it to annotate the concept "data mining". One simulated user is a data mining researcher with statistical flavor, while the other is with a combinatorial flavor. From the results, we see that for the one with statistical flavor, the extracted context indicators tend to be related to machine learning and probabilistic modeling. By contrast, for the other researcher, the patterns about "association rules", "sequential patterns", and "applications" are ranked highly. This is consistent with the fact that different researchers may have different understandings of the general concept "data mining".

So far, our experiment results have shown that SPA can generate dictionary-like annotations for frequent patterns effectively. The annotations are meaningful and can help a user to understand the annotated patterns. The results have

also shown that we may further incorporate user's prior knowledge to further contextualize the annotations and make them more meaningful to the user. In the following two experiments, we will quantitatively evaluate the performance of SPA by applying it to two interesting tasks in bioinformatics.

### 6.3 Matching Motifs and GO Terms

A challenging and promising research topic in computational biology is to predict the functions for newly discovered protein motifs which are conserved amino acid sequence patterns characterizing the function of proteins. To solve this problem, researchers have studied how to match gene ontology (GO) terms with motifs [Tao et al. 2004]. Usually, each protein sequence, which contains a number of motifs, is assigned a set of GO terms that annotate its functions. The goal of the problem is to automatically match each individual motif with those GO terms which best represent its functions. In this experiment, we formalize the problem as: Given a set of transactions $D$ (protein sequences with motifs tagged and GO terms assigned), a set $P$ of frequent patterns in $D$ to be annotated (motifs), and a set of candidate patterns $P_c$ with explicit semantics (GO terms), our goal is for $\forall p_\alpha \in P$, find $P'_c \subseteq P_c$ which best indicate the semantics of $p_\alpha$.

We used the same dataset and judgments (i.e., gold standard) as used in Tao et al. [2004]. The data has 12181 sequences, 1097 motifs, and 3761 GO terms. We also use the same performance measure as in Tao et al. [2004] (i.e., a variant of *mean reciprocal rank (MRR)* [Kantor and Voorhees 2000], notated as MRR in the following sections for convenience) to evaluate the effectiveness of the SPA technique on the motif-GO term matching problem.

Let $G = \{g_1, g_2, \ldots, g_c\}$ be a set of GO terms. Given a motif pattern $p_\alpha$, $G' = \{g'_1, g'_2, \ldots, g'_k\} \subseteq G$ is a set of "correct" GO terms for $p_\alpha$ in our judgement data. We rank $G$ with the SPA system and pick the top ranked terms, where $G$ is treated as either context units or semantically similar patterns to $p_\alpha$. This will give us a rank for each $g_i \in G$, say $r(g_i)$. MRR (with respect to $p_\alpha$) is then computed as

$$MRR_\alpha = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{r(g'_i)},$$

where $r(g'_i)$ is the $i$th correct GO term for $p_\alpha$. If $g'_i$ is not in the top ranked list, we set $1/r(g'_i) = 0$. We take the average over all motifs $\overline{MRR} = 1/m \sum_{P_\alpha \in P} MRR_\alpha$ to measure the overall performance, where $m$ is the number of motifs in our judgement file. Clearly, $0 \leq \overline{MRR} \leq 1$. A higher $\overline{MRR}$ value indicates a higher precision, and the top-ranked GO terms have the highest influence on $\overline{MRR}$, which is intuitively desirable.

If we are ranking the full candidate GO set for annotation, a "lazy" system may either just give them the same rank, or rank them randomly. It is easy to show that the expected $\overline{MRR}$ score for these two cases are the same, which is

$$E[\overline{MRR}] = \frac{1}{|G|} \sum_{i=1}^{|G|} \frac{1}{r(g_i)},$$

Table V. $\overline{MRR}$ of SPA on Motif-GO Matching

| $\overline{MRR}$ | Use MI | Use Co-occurrence |
|---|---|---|
| Context Strength | 0.5877 | 0.6064 |
| Semantical Similarity | 0.4017 | 0.4681 |
| Random ($\|G\| = 3761$) | 0.0023 | |

where $|G|$ is the number of GO terms in $G$. We note that $E[\overline{MRR}]$ drops monotonously when $|G|$ increases, which indicates that the larger the candidate set, the more difficult the ranking task. We use this value as the baseline to compare our results.

We employ all the motifs and GO terms as context units. Since these patterns are not overlapping each other, we do not use microclustering to preprocess the context units. We compare the ranking of GO terms either as context indicators or as SSPs. We also compare the use of mutual information and cooccurrence as strength weights for context units. These strategies are compared in Table V.

We see that SPA is quite effective in matching motifs with GO terms, consistently outperforming the baseline. Ranking GO terms as context units achieves better results than ranking them as SSPs, which is reasonable because a GO term usually describes only one aspect of a motif's function and is shared by a number of motifs, thus its context is likely quite different from that of a motif.

Interestingly, we notice that although mutual information is a better measure in principle, in this specific problem, using MI as the strength weight for context units is not as good as using simple cooccurrence. This may be because there are hardly many GO terms that are globally very common in this dataset, and therefore MI overpenalizes the frequent patterns. A detailed discussion on why the cooccurrence measure outperforms MI on motif-GO matching is given in Tao et al. [2004].

### 6.4 Matching Gene Synonyms

As discussed in Section 4.3, the algorithm for extracting semantically similar patterns aims at finding patterns whose meaning is very close to the pattern to be annotated. Ideally, they would be synonyms or thesauri of the given pattern. These patterns may not ever cooccur with the given pattern but tend to have similar contexts, thus cannot be extracted as strong context indicators. We do another experiment to test the performance of SPA on extracting SSPs.

In biomedical literature, it is common that different terms or aliases are used in different studies to denote the same gene. These are known as gene synonyms (see, e.g., Table VI). These synonyms generally do not appear together but are "replaceable" with each other. Detecting them can help many literature mining tasks. In this experiment, we test the application of SPA to matching gene synonyms.

We construct the synonym list for 100 fly genes which are randomly selected from the data provided by BioCreAtIvE Task 1B.[3] Ling et al. collected 22092

[3]http://www.pdg.cnb.uam.es/BioLINK/BioCreative.eval.html

Table VI. Examples of Gene Synonym Patterns

| Gene_id | Gene Synonyms |
|---|---|
| FBgn0000028 | abnormal chemosensory jump 6; acj 6; ipou; i pou; cg 9151; ti pou; twin of i pou; |
| FBgn0001000 | female sterile 2 tekele; fs 2 sz 10; tek; fs 2 tek; tekele; |

Table VII. $\overline{MRR}$ of SPA on Gene Synonym Matching

| Context Units | min sup | No Micro-Clustering | One-pass $\gamma = 0.9$ | Hierarchical $\gamma = 0.9$ |
|---|---|---|---|---|
| Closed Sequential Patterns | 0.15% | 0.5108 | 0.5161 | **0.5199** |
| | 0.18% | 0.5140 | 0.5191 | **0.5225** |
| | 0.24% | 0.5220 | 0.5245 | **0.5301** |
| | 0.3% | 0.5281 | **0.5292** | 0.5281 |
| Single Words | 0.4774 | | | |
| Random | 0.1049 ($\|G\| = 41$) | | | |

abstracts from MEDLINE[4] which contain the keyword "Drosophila" [Ling et al. 2006]. We extract the sentences from those abstracts which contain at least one synonym in the synonym list. Only the synonyms with support $\geq 3$ are kept, which gives us a small set of 41. We then mix those synonyms which belong to different genes and use the algorithm of extracting SSPs to recover the matching of synonyms. Specifically, given a synonym from the mixed list, we rank all synonyms with the SSP extraction algorithm. The performance of the system is evaluated by comparing the ranked list with the correct synonyms for the same gene. We also use MRR as the evaluation measure. The results are shown in Table VII.

From Table VII, we see that the SPA algorithm is also effective for matching gene synonyms, and significantly outperforms the random baseline. When using closed sequential patterns as context units, we always achieve better results than using single words (items), where a higher minimum support (minsup) usually yields better results. When closed sequential patterns are used, further microclustering indeed improves the performance of the system. However, when the minsup is higher, this improvement is decaying. This is reasonable because when the minsup is higher, there is less redundancy among the output closed patterns. Using hierarchical microclustering is slightly better than using the one-pass algorithm, but not always.

Finally, we discuss the performance of microclustering in removing redundant context units. The effectiveness and efficiency are shown in Figure 1. Both microclustering methods improve the precision (MRR score) when more redundant patterns are grouped into clusters. However, when $\gamma$ is set too large, the precision decreases. This indicates that we may have overpenalized the redundancy and lost useful context units. A good $\gamma$ for this task is around 0.8.

Although the cluster quality may not be optimized, the performance of one-pass microclustering is comparable to hierarchical microclustering on this task. While in principle the hierarchical clustering is not efficient, its early

[4]http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

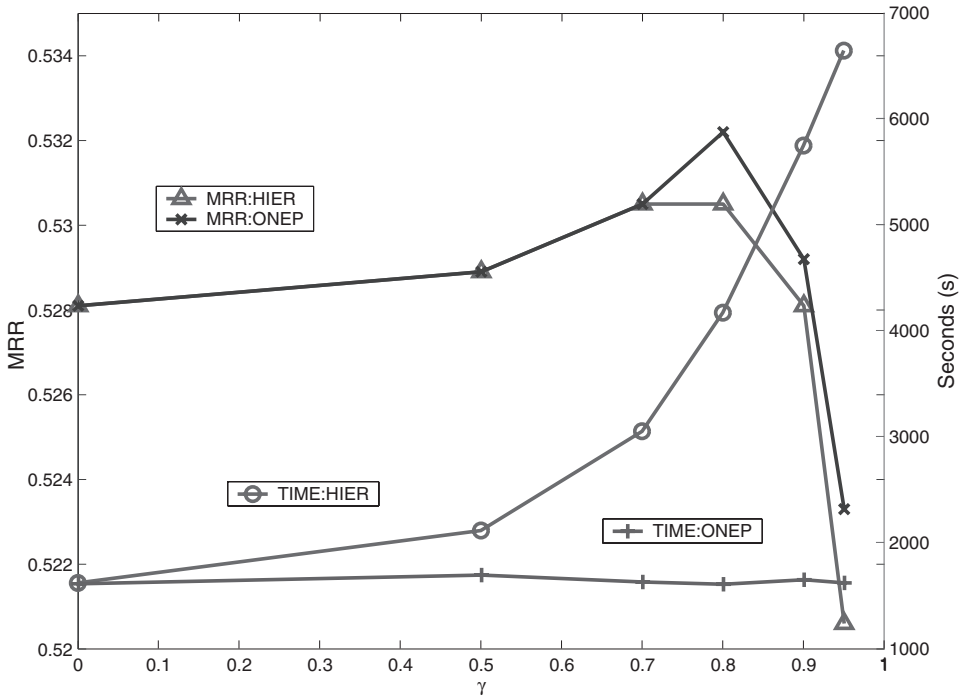


Fig. 1. Effect of microclustering algorithms. HIER: hierarchical microclustering; ONEP: one-pass microclustering; minsup = 0.3% Avg. $\gamma = 0.96$.

termination by using a small $\gamma$ saves a lot of time. The one-pass algorithm is more efficient than hierarchical clustering, and is not affected by $\gamma$. The overhead that both algorithms suffer is the computation of Jaccard distances for all pairs of patterns, namely, $O(n^2)$, where n is the number of patterns. However, this computation can be coupled in frequent pattern mining, as discussed in Xin et al. [2005].

## 7. RELATED WORK

To the best of our knowledge, the problem of semantic pattern annotation has not been well studied in existing work.

Most frequent pattern mining work [Agrawal et al. 1993; Han et al. 2004; Agrawal and Srikant 1995; Yan and Han 2002] focuses on discovering frequent patterns efficiently from the database, and does not address the problem of pattern postprocessing. To solve the problem of high redundancy in discovered patterns, closed frequent pattern [Pasquier et al. 1999], maximal frequent pattern [Bayardo 1998], and top-$k$ closed pattern [Han et al. 2002] are proposed to shrink the size of output patterns while keeping the important ones. Other researchers (e.g., DuMouchel and Pregibon [2001], Gionis et al. [2006], and Webb [2007]) have explored notions of statistical significance to extract significant patterns. However, none of this work provides additional information other than simple statistics to help users interpret the frequent patterns. The context information for a pattern tends to be ignored.

Recently, researchers have developed new techniques to approximate and summarize a frequent pattern set [Afrati et al. 2004; Yan et al. 2005], or to mine compressed frequent pattern sets [Xin et al. 2005]. Although they explored some kind of context information, none of the work can provide in-depth semantic annotations for frequent patterns as we do in our work. The context model proposed in our work covers both the pattern profile in Yan et al. [2005] and transaction coverage in Xin et al. [2005] as special cases.

Context and semantic analysis are quite common in natural language and text processing (see, e.g., Salton et al. [1975], Deerwester et al. [1990], and Lin and Pantel [2001]). Most work, however, deals with nonredundant word-based contexts, which are quite different from pattern contexts.

In specific domains, people have explored the context of specific data patterns to solve specific problems [Tao et al. 2004; Ling et al. 2006]. Although not optimally tuned, the general techniques proposed in our work can be well applied to those tasks.

## 8. CONCLUSIONS

Existing frequent pattern mining work usually generates a huge amount of frequent patterns without providing enough information to interpret the meanings of the patterns. Some recent work introduced postprocessing techniques to summarize and compress the pattern set, which shrinks the size of the output set of frequent patterns but does not provide semantic information for patterns.

In this article, we studied the novel problem of semantic pattern annotation (SPA), that is, generating semantic annotations for frequent patterns. A semantic annotation consists of a set of strongest context indicators, a set of representative transactions, and a set of semantically similar patterns (SSPs) to a given frequent pattern. We defined a general vector-space context for a frequent pattern, and proposed algorithms to exploit context modeling and semantic analysis to generate semantic annotations automatically. We also discussed how to extract supporting information for the annotations generated to further help a user to understand the annotations and meaning of an annotated pattern. We also discussed various ways of incorporating a user's prior knowledge.

The context modeling and semantic analysis method we presented is quite general and can deal with any type of frequent pattern with context information. The method can be coupled with any frequent pattern mining technique as a postprocessing step to facilitate interpretation of the discovered patterns.

We evaluated our approach on three different datasets and tasks. The results show that our methods can generate semantic pattern annotations effectively. The incorporation of a user's prior knowledge is shown to further improve the interpretability of the generated annotations. Also, as shown in our experiments, our method can be potentially applied to many interesting real-world tasks through selecting different context units and focusing on candidate patterns for SSPs.

The proposed SPA framework is quite general. However, in this article, we only studied some specific instantiation of the framework based on mutual

information weighting and the cosine similarity measure. A major goal for future research is to fully develop the potential of the proposed framework by studying alternative instantiations. For example, we may explore other options for context unit weighting and semantic similarity measurement, the two key components in our framework. It would also be very interesting to develop a probabilistic framework for annotating frequent patterns.

## ACKNOWLEDGMENTS

We thank Tao Tao and Xu Ling for providing the datasets of motif-GO matching and gene synonym matching, respectively.

## REFERENCES

AFRATI, F., GIONIS, A., AND MANNILA, H. 2004. Approximating a collection of frequent sets. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 12–19.

AGRAWAL, R., IMIELISKI, T., AND SWAMI, A. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 207–216.

AGRAWAL, R. AND SRIKANT, R. 1995. Mining sequential patterns. In *Proceedings of the 11th International Conference on Data Engineering*, 3–14.

BAYARDO, J. R. J. 1998. Efficiently mining long patterns from databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 85–93.

BRIN, S., MOTWANI, R., AND SILVERSTEIN, C. 1997. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 265–276.

BURGES, C., SHAKED, T., RENSHAW, E., LAZIER, A., DEEDS, M., HAMILTON, N., AND HULLENDER, G. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, 89–96.

CHURCH, K. W. AND HANKS, P. 1990. Word association norms, mutual information, and lexicography. *Comput. Linguist. 16*, 1, 22–29.

COVER, T. M. AND THOMAS, J. A. 1991. *Elements of Information Theory*. Wiley.

CROFT, W. B. AND LAFFERTY, J. 2003. *Language Modeling for Information Retrieval*. Kluwer.

DEERWESTER, S. C., DUMAIS, S. T., LANDAUER, T. K., FURNAS, G. W., AND HARSHMAN, R. A. 1990. Indexing by latent semantic analysis. *J. Amer. Soc. Inf. Sci. 41*, 6, 391–407.

DESHPANDE, M., KURAMOCHI, M., AND KARYPIS, G. 2003. Frequent sub-structure-based approaches for classifying chemical compounds. In *Proceedings of the International Conference on Information and Data Management (ICDM)*, 35.

DUMOUCHEL, W. AND PREGIBON, D. 2001. Empirical bayes screening for multi-item associations. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 67–76.

FANG, H., TAO, T., AND ZHAI, C. 2004. A formal study of information retrieval heuristics. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

GIONIS, A., MANNILA, H., MIELIKÄINEN, T., AND TSAPARAS, P. 2006. Assessing data mining results via swap randomization. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 167–176.

GRAHNE, G. AND ZHU, J. 2003. Efficiently using prefix-trees in mining frequent itemsets. In *FIMI'03 Workshop on Frequent Itemset Mining Implementations*.

HAN, J., PEI, J., YIN, Y., AND MAO, R. 2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov. 8,* 1, 53–87.

HAN, J., WANG, J., LU, Y., AND TZVETKOV, P. 2002. Mining top-k frequent closed patterns without minimum support. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*.

JACCARD, P. 1908. Nouvelles recherches sur la distribution florale. *Bull. Soc. Vaudoise Sci. Nat. 44*, 223C-270.

JOACHIMS, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 133–142.

KANTOR, P. AND VOORHEES, E. 2000. The TREC-5 confusion track: Comparing retrieval methods for scanned text. *Inf. Retriev. 2*, 165–176.

KLEINBERG, J. M. 1999. Authoritative sources in a hyperlinked environment. *J. ACM 46*, 5, 604–632.

KROVETZ, R. 1993. Viewing morphology as an inference process. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 191–202.

LIN, D. AND PANTEL, P. 2001. Induction of semantic classes from natural language text. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Recovery and Data Mining*, 317–322.

LING, X., JIANG, J., HE, X., MEI, Q., ZHAI, C., AND SCHATZ, B. 2006. Automatically generating gene summaries from biomedical literature. In *Proceedings of the Pacific Symposium on Biocomputing*, 40–51.

PANTEL, P. AND LIN, D. 2002. Discovering word senses from text. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 613–619.

PASQUIER, N., BASTIDE, Y., TAOUIL, R., AND LAKHAL, L. 1999. Discovering frequent closed itemsets for association rules. In *Proceeding of the 7th International Conference on Database Theory*, 398–416.

SALTON, G., WONG, A., AND YANG, C. S. 1975. A vector space model for automatic indexing. *Commun. ACM 18,* 11, 613–620.

TAO, T., ZHAI, C., LU, X., AND FANG, H. 2004. A study of statistical methods for function prediction of protein motifs. *Appl. Bioinf. 3,* 2-3, 115–124.

WAGNER, R. A. AND FISCHER, M. J. 1974. The string-to-string correction problem. *J. ACM 21,* 1, 168–173.

WANG, K., XU, C., AND LIU, B. 1999. Clustering transactions using large items. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)*, 483–490.

WEBB, G. I. 2007. Discovering significant patterns. *Mach. Learn. 68,* 1, 1–33.

XIN, D., HAN, J., YAN, X., AND CHENG, H. 2005. Mining compressed frequent-pattern sets. In *Proceedings of VLDB International Conference on Very Large Databases*, 709–720.

YAN, X., CHENG, H., HAN, J., AND XIN, D. 2005. Summarizing itemset patterns: A profile-based approach. In *Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 314–323.

YAN, X. AND HAN, J. 2002. gspan: Graph-Based substructure pattern mining. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 721–724.

YAN, X., HAN, J., AND AFSHAR, R. 2003. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM)*, 166–177.

Received November 2006; revised August 2007; accepted September 2007