Generating Semantic Annotations for Frequent Patterns with Context Analysis

Qiaozhu Mei, Dong Xin, Hong Cheng, Jiawei Han, ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana Champaign
Urbana,IL 61801
{ qmei2, dongxin, hcheng3, hanj, czhai }@uiuc.edu

ABSTRACT

As a fundamental data mining task, frequent pattern mining has widespread applications in many different domains. 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 some recent work has studied the compression and summarization of frequent patterns, the proposed techniques can only annotate a frequent pattern with non-semantical information (e.g. support), which provides only limited help for a user to understand the patterns.

In this paper, we propose the novel problem of generating semantic annotations for frequent patterns. The goal is to annotate a frequent pattern with in-depth, concise, and structured information that can better indicate the hidden meanings of the pattern. 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 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

Keywords: frequent pattern, pattern annotation, pattern context, pattern semantic analysis

1. INTRODUCTION

With its broad applications such as association rule mining [2], correlation analysis [4], classification [6], and clustering [19], discovering frequent patterns from large databases has been a central research topic in data mining for years. Various techniques have been developed for mining frequent item sets [2, 8], sequential patterns [3], graph patterns [22], etc. These techniques can usually output a large, complete set of frequent patterns efficiently, and provide basic statistic information such as support for each pattern. However, the excessive volume of the output pattern set and the lack of context information has made it difficult to interpret and explore the patterns. In most cases, a user only wants to explore a small set of most interesting patterns, and before exploring them, to have a rough idea about their hidden meanings or 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 the main ideas of the paper. Similarly, it is also highly desirable to have such a summary for a frequent pattern to explain or indicate the potential meanings of the pattern and to help a user decide whether and how to explore the pattern. Therefore, a new major challenge in frequent pattern mining has been raised by researchers, which is how to present and interpret the patterns discovered, in order to 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 [15], and maximum frequent pattern [16] to shrink the size of output patterns and provide more information beyond "*support*". Recently, novel methods have been proposed either to mine a compressed set of frequent patterns [20] or to summarize a large set of patterns with the most representative ones [21]. Both of them employ extra information of frequent patterns beyond the simple information of *support*, which is either transaction coverage [20] or pattern profiles [21]. These methods can successfully reduce the number of output patterns and present only the most interesting ones to the user. However, the information that these methods use to annotate a frequent patten is restricted to the morphological information or simple statistics (e.g. support, transaction coverage, profile); from such an annotation, users could not infer the semantics, or hidden meanings of the pattern, thus still have to look through all the data transactions in which a pattern occurs in order to figure out whether it is worth exploring.

In this paper, we study the problem of automatically generating semantic annotations for frequent patterns, by which we mean to extract and provide concise and in-depth information for a frequent pattern, which indicates the semantics,

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

or hidden meanings, of the pattern.

What is an appropriate semantic annotation for a frequent pattern? Generally, the hidden meaning of a pattern can be inferred from the patterns with similar meanings, the data objects co-occurring with it, and the transactions in which the pattern appears. In principle, we expect such an annotation to be compact, well structured, and indicative of the meanings of the pattern. This criterion is analogous to dictionary entries, which annotate each term with structured semantic information.

Example 1: An example of a dictionary entry[1]

Dictionary Term: "*pattern*"
['pætən], 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 non-semantic information is presented. Second, a group of definitions are given, which suggests the semantics of the term, followed by several example sentences, which show the usage of this term in context. Besides, a set of synonyms, thesaurus or semantically similar terms are presented, which have similar definitions with this term.

Analogically, 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 further explore it. Given a frequent pattern, it is trivial to generate non-semantic information such as basic statistics and morphological information, so the main challenge is to generate the semantic descriptions of a pattern, which is the goal of our work. 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 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 [13]. Specifically, by defining and analyzing the context of a pattern, we can find strong context indicators and use them to represent the meanings of a pattern. Second, we also want to extract the data transactions that best represent the meanings of the pattern, which is analogical to the example sentences. Finally, semantically similar patterns (SSPs) of the given pattern, i.e., patterns with similar contexts as the original pattern, can be extracted and presented. This is similar to the synonyms or thesauri of a term in dictionary. Therefore, an example of semantic pattern annotation (SPA) can be shown as follows:

Example 2: An example of annotating a frequent pattern

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

The term "frequent pattern" in this example is itself a frequent itemset, or a 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.

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 experiments 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 paper 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. We discuss our experiments and results in Section 5, the related work in Section 6, and our conclusions in Section 7, respectively.

2. PROBLEM FORMULATION

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

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

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

Definition 2 (Context Unit): Given a dataset D and

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

the set of frequent patterns P_D, a *context unit* is a basic object in D which carries semantic information and co-occurs 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, a pattern, or a transaction in practice, 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_α, denoted as $c(\alpha)$, is represented by a selected set of context units $U_\alpha \subseteq U_D$ such that every $u \in U_\alpha$ co-occurs with p_α. Each selected context unit u is also called a ***context indicator*** of p_α, associated with a *strength weight* $w(u, \alpha)$, which measures how well it indicates the semantics of p_α.

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: An example of 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, ...
An exemplary frequent pattern ($\sigma = 0.5$)
p = {diaper, milk}
Context indicators of p**:**
diaper, baby carriage, {milk, baby lotion}, t_1, ...

With the definitions above, we now define the concept of *semantic annotation* for a frequent pattern and the related 3 subproblems.

Definition 4 (Semantic Annotation): Let p_α be a frequent pattern in a dataset D, U_α be the set of context indicators of p_α, and P be a set of patterns in D. A *semantic annotation* of p_α consists of: 1) a set of context indicators of p_α, $I_\alpha \subseteq U_\alpha$, s.t. $\forall u \in I_\alpha$ and $\forall u' \in U_\alpha - I_\alpha$, $w(u', \alpha) \leq w(u, \alpha)$; 2) a set of transactions $T_\alpha \subseteq D_\alpha$, s.t.$\forall t \in T_\alpha$ 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 $P' \subseteq P$ s.t. $\forall p \in P'$ 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_α.

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_α, s.t.$\forall t \in T_\alpha$ 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_α, s.t. $\forall p \in P'$ 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_α.

With the definitions above, we may define the 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, and for a transaction and a pattern context;
3) for a given frequent pattern, extract the most significant context indicators, representative transactions and semantically similar patterns to construct a structured annotation.

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 is 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 – they do not depend on any domain knowledge about the dataset or the 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. MODELING PATTERN CONTEXTS

In this section, we discuss how to model pattern contexts 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.

Vector Space Model (VSM) [17] 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 (i.e., 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 the vector dimensions and gives the most flexibility to the selection of dimensions and weights. Formally, the context of a frequent pattern is modeled as follows.

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

The two key issues in a VSM are to select the vector dimensions and to assign weights for each dimension [17]. Specifically, the effectiveness of context modeling is highly dependent on how to select context units and design the strength weights. Actually, due to the generality of VSM, the proposed vector-space pattern context model is quite

general and covers different strategies for context unit selection and weighing functions. In the following subsections, we first discuss the generality of the context model, and then discuss specific solutions for the two issues respectively.

3.1 The Generality of Context Modeling

Some existing work has explored non-morphological information of frequent patterns with some concepts related to the "pattern context" defined above. We now show that the notion of "pattern context" is more general and can cover those concepts as special cases.

In [21], Yan et al. introduced the profile of an itemset for summarizing itemset patterns, which is represented as a Bernoulli Distribution Vector. In fact, this "profile" of a frequent itemset α can be written as a vector $\langle w(o_1, \alpha), w(o_2, \alpha), ..., w(o_d, \alpha)\rangle$ over all the single items $\{o_i\}$ in D. Here $w(o_i, \alpha) = \frac{\sum_{t_j \in D\alpha} t_j^i}{|D_\alpha|}$, 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 defined, where single items are selected as context units.

Xin and others proposed a distance measure for mining compressed frequent-pattern sets, which is computed based on the transaction coverage of two patterns [20]. Interestingly, the "transaction coverage" is also a specific instance of "pattern context". Given a frequent pattern p_α, the transaction coverage of p_α can be written as a vector $\langle w(t_1, \alpha), w(t_2, \alpha), ..., w(t_k, \alpha)\rangle$ over all the transactions $\{t_i\}$ in D, where each transaction is selected as a context unit, and $w(t_i, \alpha) = 1$ if $p_\alpha \in t_i$ and 0 otherwise.

Covering the concepts in existing work as specific instances, the pattern context model we proposed is general and has quite a few benefits. First, it does not assume pattern types. The pattern profile proposed in [21] assumes that both transactions and patterns are itemsets, thus does not work for other patterns such as sequential patterns and graph patterns. Second, the 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 introduced bridges this gap by allowing various granularity of semantic units, and allows the user to explore the pattern semantics at the level that corresponds to their 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.

3.2 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 a **pattern** is the minimal units which carries semantic information in a dataset, and thus 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 single item can be viewed as a sequential pattern of length 1, and every transaction can be viewed as a sequential pattern which is identical to the transactional sequence. The choosing of patterns as context units is task dependent, and can usually be optimized with prior knowledge about the task and the data. For example, we can use words as context units in a text database, and 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 granularity may cause the 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 discriminative as possible to indicate the meanings of a pattern. However, when various granularity of patterns are selected as context units, some units will become less discriminative, and more severely, some becomes redundant. For example, when the pattern "mining subgraph" is added as a context unit, the discriminating power of other units like "mining frequent subgraph" and "subgraph" would 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 discriminative to indicate the semantics of the frequent patterns co-occurring 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 the semantic discriminating power.

3.2.1 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 patterns in principle, we are practically not interested 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 [21], but they only work for itemset patterns and are not general enough for our purpose.

Some techniques such as LSI [5] 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, if d_1 and d_2 correspond to the patterns "AB" and "ABC" respectively, and if we consider d_2 to be redundant w.r.t d_1, we do not expect the information of d_2 to be preserved after the removal of d_2.

3.2.2 Redundancy Removal: Closed Frequent Pattern

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 sub patterns, we explore closed frequent patterns [15] and maximum frequent patterns [16] to solve this problem.

A maximal frequent pattern is a frequent pattern which does not have a frequent super-pattern. It is easy to show that maximum frequent pattern is not appropriate for this problem since it may lose important discriminative units. For example, the frequent pattern "data cube", although not a maximum frequent pattern, 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_α is closed if and only if there exists no super-pattern p_β of p_α, s.t. $D_\alpha = D_\beta$.

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_α 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_β as a representative of p_α and p_β 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 [15, 23].

3.2.3 Redundancy Removal: Microclustering

However, as stated in [21], 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. Those 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 the 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 microclutering algorithms to further group the close frequent patterns.

Definition 9 (Jaccard Distance): Let p_α and p_β as two frequent patterns. The Jaccard Distance between p_α and p_β is computed as:

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

Jaccard Distance [10] is commonly applied to cluster data based on their co-occurrence in transactions. Our need is to group the 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 [20].

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:

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 are 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 γ. The second algorithm, which we call One-Pass Microclustering, iteratively assigns a closed frequent pattern p_α to its nearest cluster if the distance is below γ, where the distance between p_α and a cluster C is defined as the Jaccard distance between p_α 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 γ. Only the medoid of each cluster is selected as a context unit. By varying γ, 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 are compared in Section 5.

Algorithm 1 Hierarchical Microclustering

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

initialize n clusters C_i, each as a closed frequent pattern;
compute the Jaccard Distance d_{ij} among $\{p_1, ..., p_n\}$;
set the current minimal distance $d = min(d_{ij})$;
while $(d < \gamma)$
  select d_{st} where $(s, t) = \text{argmin}_{i,j} d_{ij}$;
  merge clusters C_s and C_t into a new cluster C_u;
  foreach $C_v \neq C_u$
    compute $d_{uv} = max(d_{\alpha\beta})$ where $p_\alpha \in C_u, p_\beta \in C_v$;
foreach C_u;
  foreach $p_\alpha \in C_u$;
    compute $\bar{d}_\alpha = avg(d_{\alpha\beta})$ where $p_\beta \in C_u$;
  add p_α into $\mathcal{P}'$, where $\alpha = \text{argmin}_i(\bar{d}_i)$;
return

Algorithm 2 One-pass Microclustering

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

initialize 0 clusters;
compute the Jaccard Distance d_{ij} among $\{p_1, ..., p_n\}$;
foreach $(p_\alpha \in \mathcal{P})$
  foreach cluster C_u
    $\tilde{d}_{\alpha,u} = max(d_{\alpha\beta})$ where $p_\beta \in C_u$;
  $v = \text{argmin}_u(\tilde{d}_{\alpha,u})$;
  if$(\tilde{d}_{\alpha,v} < \gamma)$
    assign p_α to C_v
  else
    initialize a new cluster $C = \{p_\alpha\}$
foreach C_u;
  foreach $p_\alpha \in C_u$;
    compute $\bar{d}_\alpha = avg(d_{\alpha\beta})$ where $p_\beta \in C_u$;
  add p_α into $\mathcal{P}'$, where $\alpha = \text{argmin}_i(\bar{d}_i)$;
return

3.3 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, which represents how strong the context unit corresponding to this dimension indicates the meaning of a given pattern. Intuitively, the strongest context indicators for a pattern p_α should be those units that frequently co-occur with p_α but infrequently co-occur with others.

Practically, many types of weighting functions can be used to measure the strength of a context indicator. For example, we can assign the weight for a context indicator u for p_α as the number of transactions with both u and p_α. However, in principle, a good weighting function is expected to satisfy several **constraints**:

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

1. $w(u_i, p_\alpha) \leq w(p_\alpha, p_\alpha)$: the best semantic indicator of p_α 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_α is independent: u_i cannot indicate the semantics of p_α.

An obvious choice is co-occurrences, which however, may not be a good measure. One one hand, it does not satisfy constraints 3. On the other hand, we want to penalize the context units that are globally common patterns in the collection. Which means, although they may co-occur many times with p_α, it may still not be a good context indicator for p_α because it also co-occurs frequently with others. In general, the context units that are strongly correlated to p_α should be weighted higher. In our work, we introduce a more principled measure.

Mutual Information (MI) is widely used to measure the mutual independency of two random variables in information theory, which intuitively measures how much information a random variable tells about the other. The definition of mutual information is given as

Definition 10: (Mutual Information). Given two frequent patterns p_α and p_β, let $X = \{0, 1\}$ and $Y = \{0, 1\}$ be two random variables for the appearance of p_α and p_β 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 the three constraints 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 $U = \{u_1, u_2, ..., u_k\}$ be a selected set of k context units and $w(\cdot, p_\alpha)$ be the unit weighting function w.r.t. any frequent pattern p_α, i.e. $I(\cdot; p_\alpha)$. The context model, or context vector $c(\alpha)$ for p_α is $\langle w(u_1, p_\alpha), w(u_2, p_\alpha), ..., w(u_k, p_\alpha)\rangle$.

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

Definition 11 (Semantical Similarity): Let p_α, p_β, p_δ 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 $sim(c(\alpha), c(\beta)) > sim(c(\alpha), c(\delta))$, we say that p_β is semantically more similar to p_α than p_δ w.r.t. $sim(c(\cdot), c(\cdot))$.

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 [17]. In our work, we use cosine similarity of two context vectors to measure the semantic similarity of two corresponding frequent patterns. 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 $c(\alpha) = \langle a_1, a_2, ..., a_k\rangle$ and $c(\beta) = \langle b_1, b_2, ..., b_k\rangle$.

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

4.1 Extracting Strongest Context Indicators

Let p_α be a frequent pattern and $c(\alpha)$ be its context model, which is defined in this work as a context vector $\langle w_1, w_2, ..., w_k\rangle$ over a set of context units $U = \{u_1, u_2, ..., u_k\}$. As defined in Section 2, w_i is a weight for u_i which tells how well u_i indicates the semantics of p_α. 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 $w(\cdot, p_\alpha)$, e.g., 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.

4.2 Extracting Representative Transactions

Let p_α be a frequent pattern, $c(\alpha)$ be its context model, and $D = \{t_1, ...t_l\}$ be a set of transactions, our goal is to select k_t transactions $T_\alpha \subseteq D$ with a similarity function $s(\cdot, p_\alpha)$, s.t. $\forall t \in T_\alpha$ 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_α, i.e., over $\{u_1, u_2, ..., u_k\}$. Then, we use the cosine similarity presented in Section 3 to compute the similarity between a transaction t and the context of p_α. The rest is again a ranking problem. Formally, let $c(t) = \langle w'_1, w'_2, ..., w'_k\rangle$ 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_α be a frequent pattern, $c(\alpha)$ be its context model, and $P_c = \{p_1, ..., p_c\}$ be a set of frequent patterns which are believed to be good candidates for annotating the semantics of p_α, i.e., 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_α. Formally, let $\{c(p_1), ..., c(p_c)\}$ be the context vectors for $\{p_1, ..., p_c\}$. 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_α, or a set of heterogenous patterns. For example, it can be a set of patterns or terminology from the domain that a user is familiar with, and is used to annotate patterns from an unfamiliar domain. This brings great flexibility to apply the 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 5.

5. 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.

5.1 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 co-authorship, each 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. Our experiments are designed as follows:

1) Given a set of authors/co-authors, annotate each of them with their strongest context indicators, the most representative titles from their publications, and the co-authors or title patterns which are most semantically similar to them. 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' work.

2) Given a set of title terms (sequential patterns), annotate each of them with their strongest context indicators, the most representative titles, the most similar terms, and the most representative author/co-authors. Note again that the most representative author/co-authors 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 [7] and CloSpan [23] to generate closed frequent itemsets of co-authors and closed sequential patterns of title terms respectively. The title words are stemmed by Krovertz stemmer [12], which converts the morphological variations of each English word to its root form. We set the minimum support for frequent itemset 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 those sequential patterns and get a smaller set of 3443 patterns, with $\gamma = 0.9$ (the average Jaccard distance between these patterns is > 0.95).

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

Table 1: Effectiveness of Microclustering

Table 1 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 pattern "data mine" and "mine data" are assigned to different clusters, which cannot be achieved by the existing pattern summarization techniques such as [21]. The results generated by hierarchical microclustering are similar.

In Table 2, 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 co-authorship or single authors, the strongest context indicators are usually their other co-authors 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 co-occur with the given pattern in a paper. For example, the pattern "jiayong_wang", "jiong_yang&philip_s_yu&wei_wang" actually do not co-occur with the pattern "xifeng_yan&jiawei_han", but are extracted because their contexts are similar. For a single author, whose context is usually more diverse, the SSPs are more likely to be title terms instead of authors.

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 tend to co-appear with them. Their SSPs usually provide interesting concepts or descriptive terms which are close to their meanings, e.g. "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 2 for each transaction.

These experiments show that the SPA can generate dictionary like annotations for frequent patterns effectively. In the following two experiments, we quantitatively evaluate the performance of SPA, by applying it to two interesting tasks.

5.2 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

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

Pattern	Type	Annotations
xifeng_yan	I	graph; philip_s_yu; mine close; mine close frequent; index approach; graph pattern; sequential pattern
jiawei_han	T	gspan graph-base substructure pattern mine
	T	mine close relational graph connect constraint
(SSP set =	T	clospan mine close sequential pattern large database
co-author patterns)	S	jiawei_han&philip_s_yu; jian_pei&jiawei_han; jianyong_wang; jiong_yang&philip_s_yu&wei_wang
	I	spiros_papadimitriou; fast; use fractal; graph; use correlate;
christos_faloutso	T	multiattribute hash use gray code
(SSP set =	T	recovere latent time-sery their observe sum network tomography particle filter
title term patterns)	T	index multimedia database tutorial
	S	use fractal; fast data mine; data graph; efficient time sequence; spatial access method; discovery correlate
information	I	w_bruce_croft; web information; monika_rauch_henzinger; james_p_callan; full-text;
retrieval	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;

Table 2: Annotations Generated for Frequent Patterns in DBLP Dataset
Note: "I" means context indicators; "T" means representative transactions; "S" means semantically similar patterns. We exclude 12 most frequent and non-informative English words from the collection when extracting frequent patterns.

Ontology(GO) terms with motifs [18]. 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 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_α.

We used the same data set and judgments (i.e., gold standard) as used in [18]. The data has 12181 sequences, 1097 motifs, and 3761 GO terms. We also use the same performance measure as in [18] (i.e., a variant of **Mean reciprocal rank (MRR)** [11], 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 $G = \{g_1, g_2, ..., g_c\}$ be a set of GO terms. Given a motif pattern p_α, $G' = \{g'_1, g'_2, ..., g'_k\} \subseteq G$ is a set of "correct" GO terms for p_α 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_α. This will give us a rank for each $g_i \in G$, say $r(g_i)$. MRR (w.r.t. p_α) 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_α. If g'_i is not in the top ranked list, we set $1/r(g'_i) = 0$. We take the average over all the 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

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

where $|G|$ is the number of GO terms in G. $E[\overline{MRR}]$ drops monotonously when $|G|$ increases, which indicates the larger the candidate set is, the more difficult is 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 with 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 co-occurrence as strength weight for context units. These strategies are compared in Table 3:

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

Table 3: $\overline{MRR}$ of SPA on Motif-GO matching

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 for the strength weight in principle, in this specific problem, using MI as strength weight for context units is not as good as using simple co-occurrence. This may be because there are hardly many Go terms that are globally very common in this dataset, and therefore MI over penalizes the frequent patterns. A detailed discussion on why co-occurrence measure outperforms MI on Motif-GO matching problem is given in [18].

5.3 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 co-occur 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, which are known as gene synonyms (see e.g., Table 4). 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.

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 4: Examples of gene synonym patterns

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 abstracts from MEDLINE[4] which contain the keyword "Drosophila" [14]. We extract the sentences from those abstracts which contain at least one synonym in the synonym list. Only the synonyms with support ≥ 3 are kept, which gives us a small set of 41 synonyms. 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 as follows.

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 ($\|G\| = 41$)			

Table 5: $\overline{MRR}$ of SPA on gene synonym matching

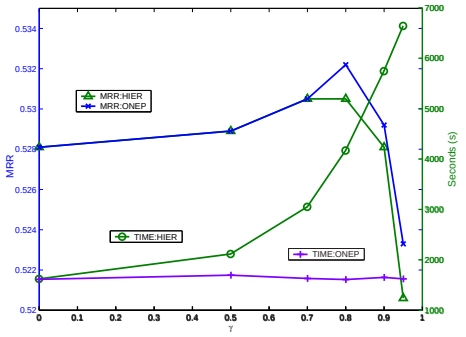

Figure 1: Effect of microclustering algorithms

HIER: hierarchical microclustering; ONEP: one-pass microclustering; minsup = 0.3% Avg. $\gamma = 0.96$;

From Table 5, we see that the SPA algorithm is also effective for matching gene synonyms, which significantly outperforms the random baseline. When using closed sequential patterns as context units, we always achieve better results than using single words (items) as context units, 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 γ is set too large, the precision decreases. This indicates that we may have over penalized the redundancy and lost useful context units. A good γ 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, the early termination by using a small γ saves a lot of time. The one-pass algorithm is more efficient than the hierarchical clustering, and is not affected by γ. The overhead that both algorithms suffer is the computation of Jaccard distances for all pairs of patterns, i.e., $O(n^2)$ where n is the number of patterns. However, this computation can be coupled in frequent pattern mining, as discussed in [20].

6. 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 [2, 8, 3, 22] 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 patterns discovered, closed frequent pattern [15], maximum frequent pattern [16] and top-k closed pattern [9] are proposed to shrink the size of output patterns while keeping the important ones. 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 develop new techniques to approximate, summarize a frequent pattern set [1, 21], or mine compressed frequent pattern sets [20]. 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 [21] and transaction coverage in [20] as special cases.

Context and semantic analysis are quite common in natural language and text processing (see e.g., [17, 5, 13]). Most work, however, deals with non-redundant 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 [18, 14]. Although not optimally tuned, the general techniques proposed in our work can be well applied to those tasks.

7. 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.

[3]http://www.pdg.cnb.uam.es/BioLINK/BioCreative.eval.html
[4]http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

We propose the novel problem of semantic pattern annotation (SPA) – 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 define a general vector-space context for a frequent pattern. We propose algorithms to exploit context modeling and semantic analysis to generate semantic annotations automatically. The context modeling and semantic analysis method we presented is quite general and can deal with any types of frequent patterns with context information. The method can be coupled with any frequent pattern mining techniques as a postprocessing step to facilitate interpretation of the discovered patterns.

We evaluated our approach on three different dataset and tasks. The results show that our methods can generate semantic pattern annotations effectively. 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.

Although the proposed SPA framework is quite general, in this paper, we only studied some specific instantiation of the framework based on mutual information weighting and 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.

8. ACKNOWLEDGMENTS

We thank Tao Tao and Xu Ling for providing the datasets of Motif-GO matching and gene synonym matching, respectively. This work was in part supported by the National Science Foundation under award numbers 0425852.

9. REFERENCES

[1] F. Afrati, A. Gionis, and H. Mannila. Approximating a collection of frequent sets. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 12–19, 2004.

[2] R. Agrawal, T. Imieliski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216, 1993.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 3–14, 1995.

[4] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: generalizing association rules to correlations. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 265–276, 1997.

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

[6] M. Deshpande, M. Kuramochi, and G. Karypis. Frequent sub-structure-based approaches for classifying chemical compounds. In *Proceedings of ICDM'03*, page 35, 2003.

[7] G. Grahne and J. Zhu. Efficiently using prefix-trees in mining frequent itemsets. In *FIMI'03 Workshop on Frequent Itemset Mining Implementations.*, 2003.

[8] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.*, 8(1):53–87, 2004.

[9] J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *Proceedings of ICDM'02*, 2002.

[10] P. Jaccard. Nouvelles recherches sur la distribution florale. *Bull. Soc. Vaudoise Sci. Nat.*, 44:223C–270, 1908.

[11] P. Kantor and E. Voorhees. The TREC-5 confusion track: Comparing retrieval methods for scanned text. *Information Retrieval*, 2:165–176, 2000.

[12] R. Krovetz. Viewing morphology as an inference process. In *Proceedings of SIGIR '93*, pages 191–202, 1993.

[13] D. Lin and P. Pantel. Induction of semantic classes from natural language text. In *Proceedings of KDD'01*, pages 317–322, 2001.

[14] X. Ling, J. Jiang, X. He, Q. Mei, C. Zhai, and B. Schatz. Automatically generating gene summaries from biomedical literature. In *Proceedings of Pacific Symposium on Biocomputing*, pages 40–51, 2006.

[15] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proceeding of the 7th International Conference on Database Theory*, pages 398–416, 1999.

[16] J. Roberto J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, 1998.

[17] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[18] T. Tao, C. Zhai, X. Lu, and H. Fang. A study of statistical methods for function prediction of protein motifs. *Applied Bioinformatics*, 3(2-3):115–124, 2004.

[19] K. Wang, C. Xu, and B. Liu. Clustering transactions using large items. In *Proceedings of CIKM'99*, pages 483–490, 1999.

[20] D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent-pattern sets. In *Proceedings of VLDB'05*, pages 709–720, 2005.

[21] X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: a profile-based approach. In *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 314–323, 2005.

[22] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *Proceedings ICDM'02*, pages 721–724, 2002.

[23] X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings of SDM'03*, pages 166–177, 2003.