Diverging Patterns: Discovering Significant Frequency Change Dissimilarities in Large Databases

Aijun An, Qian Wan, Jiashu Zhao and Xiangji Huang
Department of Computer Science and Engineering
York University, Toronto, Canada
aan, qwan, jessie, jhuang@cse.yorku.ca

ABSTRACT

In this paper, we present a framework for mining *diverging patterns*, a new type of contrast patterns whose frequency changes significantly differently in two data sets, e.g., it changes from a relatively low to a relatively high value in one dataset, but from high to low in the other. In this framework, a measure called *diverging ratio* is defined and used to discover diverging patterns. We use a four-dimensional vector to represent a pattern, and define the pattern's diverging ratio based on the angular difference between its vectors in two datasets. An algorithm is proposed to mine diverging patterns from a pair of datasets, which makes use of a standard frequent pattern mining algorithm to compute vector components efficiently. We demonstrate the effectiveness of our approach on real-world datasets, showing that the method can reveal novel knowledge from large databases.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database applications – *data mining*; I.5.2 [Pattern Recognition]: Design methodology – *pattern analysis*

General Terms: Experimentation

Keywords: Diverging patterns, contrast patterns

1. INTRODUCTION

Contrast patterns are those patterns that capture important and significant differences or changes between two or more sets of data in the same domain. Such patterns are useful for identifying distinguishing characteristics of data and can also be used to build powerful classifiers [10, 7]. A number of promising methods have been proposed to discover contrast patterns, including emerging pattern mining [6, 2], contrast set mining [4], and some other methods for mining complex types of contrasts [11, 14, 12]. However, most of the methods use support and/or confidence measures to evaluate the differences between datasets and do not consider the timestamps associated with the occurrences of a pattern. Thus, they cannot discover important differences in a pattern's temporal trends in different data sets. Let's see the following example.

Table 1: Simple gene expression data sets

	normal				*disease*			
	G_1	G_2	G_3	G_4	G_1	G_2	G_3	G_4
$t01$	1	0	1	0	1	1	1	0
$t02$	0	1	1	1	0	1	1	1
$t03$	1	1	1	0	0	1	1	0
$t04$	1	1	1	1	1	0	0	1
$t05$	1	1	0	1	1	1	1	1
$t06$	1	1	1	1	1	0	0	1
$t07$	0	1	0	1	1	1	1	1
$t08$	1	1	0	1	1	1	1	0
$t09$	1	1	0	0	1	0	1	1
$t10$	1	1	1	1	0	0	1	0
$t11$	0	1	1	0	0	1	1	1
$t12$	1	1	1	1	1	1	0	1
$t13$	1	1	0	0	1	1	1	1
$t14$	1	1	1	1	0	1	0	1
$t15$	1	0	0	1	1	1	1	1
$t16$	1	1	0	0	1	1	0	0
$t17$	0	1	1	0	1	1	1	0
$t18$	1	0	0	1	1	1	1	1
$t19$	0	1	1	0	1	0	1	1
$t20$	1	0	0	1	1	1	1	0

Suppose we have a pair of simplified gene expression data sets, *normal* and *disease*, which record the expression levels of four genes $(G_1, ..., G_4)$ in normal and diseased tissues during a biological process over a series of time-stamps $(t_{01}, ..., t_{20})$, as shown in Table 1. Each cell represents the expression level of a certain gene at a certain time-stamp, whose value is represented by 0 or 1. If a gene is up or down regulated at a specific time-stamp, the corresponding element takes a value of 1; otherwise, the value is 0. Let us consider the gene set $\{G_1, G_2\}$. Because its frequency is exactly the same ($\frac{11}{20} = 0.55$) in both data sets, it cannot be considered as a contrast pattern based on the support-confidence framework. However, interesting differences of the gene set between these two data sets can be found after we take into account the time information in the data sets. We can easily see that before (and including) $t10$ in the *normal* data set, $\{G_1, G_2\}$ appears 7 times; but after $t10$, $\{G_1, G_2\}$ only occurs 4 times. That is to say that the frequency of $\{G_1, G_2\}$ in *normal* decreases significantly from 70% to 40% after $t10$. On the contrary, the frequency of $\{G_1, G_2\}$ in *disease* increases significantly from 40% to 70%.

This observation asserts that a pattern might have significantly different distributions between two data sets, even though it has a subtle frequency-difference in the data sets. This motivates our research in this paper to identify a new class of contrast patterns to capture their significant dissimilarities between the data sets, especially those patterns whose frequency changes in opposite directions, e.g., it changes from a relatively low to a relatively high value in one data set, but from high to low in the other.

CIKM'09, November 2–6, 2009, Hong Kong, China.

The main contributions of this paper are as follows. First, we propose a framework for mining *diverging patterns*, a new type of contrast patterns whose frequency changes in different directions in a pair of data sets. Second, we define a measure, called *diverging ratio*, to capture the frequency-change-difference of a pattern in two data sets. The measure uses a four-dimensional vector to represent a pattern in a data set and is defined based on the angular difference between its vectors in two data sets. Third, an algorithm is proposed to mine diverging patterns from a pair of data sets. The algorithm makes use of a standard frequent pattern mining algorithm to compute relevant vectors efficiently. Finally, we perform experiments on real-world data sets to show the effectiveness of mining diverging patterns.

2. DIVERGING PATTERN

Let $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ be a set of m distinct *items*. A subset $X \subseteq \mathcal{I}$ is called an *itemset* or a *pattern*. A transaction over $\mathcal{I}$ is a couple $\mathcal{T} = (tid, I)$ where tid is the transaction identifier (or time-stamp) and $I \subseteq \mathcal{I}$ is an itemset. A transaction $\mathcal{T} = \langle tid, I \rangle$ is said to support an itemset $X \subseteq \mathcal{I}$, if and only if $X \subseteq I$. A transaction database $\mathcal{D}$ over $\mathcal{I}$ is a set of transactions over $\mathcal{I}$.

The *cover* of an itemset X in $\mathcal{D}$, denoted as $cov(X, \mathcal{D})$, consists of the set of transactions in $\mathcal{D}$ that support X, i.e., $cov(X, \mathcal{D}) = \{\langle tid, I\rangle \mid \langle tid, I\rangle \in \mathcal{D}, X \subseteq I\}$. An itemset X in a transaction database $\mathcal{D}$ has a *support*, denoted as $sup(X, \mathcal{D})$, which is the proportion of transactions in $\mathcal{D}$ containing X. That is, $sup(X, \mathcal{D}) = \frac{|cov(X,\mathcal{D})|}{|\mathcal{D}|}$, where $|\mathcal{S}|$ is the cardinality of set $\mathcal{S}$.

2.1 Definition of weight

Definition 1. Assuming that the transactions in a transaction database $\mathcal{D}$ are ordered by their transaction identifiers (or time-stamps), the *position* of a transaction $\mathcal{T}$ in $\mathcal{D}$, denoted as $\rho(\mathcal{T}, \mathcal{D})$, is the number of transactions whose transaction identifier (or time-stamp) is less than or equal to that of $\mathcal{T}$. Thus, $1 \leq \rho(\mathcal{T}, \mathcal{D}) \leq |\mathcal{D}|$.

Below we define the *weight* of a pattern in a database to capture the general distribution of the pattern in the database.

Definition 2. The *weight* of a pattern X in a database $\mathcal{D}$, denoted as $\overrightarrow{\omega}(X, \mathcal{D})$, is defined as a four-dimensional vector:

$$\overrightarrow{\omega}(X, \mathcal{D}) = \langle w_1(X, \mathcal{D}), w_2(X, \mathcal{D}), w_3(X, \mathcal{D}), w_4(X, \mathcal{D})\rangle,$$

where $w_1(X, \mathcal{D})$ is the sum of the normalized distances from the transactions containing X in the first half of $\mathcal{D}$ to the beginning of $\mathcal{D}$, which is defined as:

$$w_1(X, \mathcal{D}) = \frac{\sum_{\mathcal{T} \in cov(X,\mathcal{D})}\{\rho(\mathcal{T}, \mathcal{D}) | \rho(\mathcal{T}, \mathcal{D}) < \frac{|\mathcal{D}|+1}{2}\}}{|\mathcal{D}|};$$

$w_2(X, \mathcal{D})$ is the sum of the normalized distances from the transactions containing X in the first half of $\mathcal{D}$ to the middle position of $\mathcal{D}$, which is defined as:

$$w_2(X, \mathcal{D}) = \frac{\sum_{\mathcal{T} \in cov(X,\mathcal{D})}\{\lceil \frac{|\mathcal{D}|+1}{2} \rceil - \rho(\mathcal{T}, \mathcal{D}) | \rho(\mathcal{T}, \mathcal{D}) < \frac{|\mathcal{D}|+1}{2}\}}{|\mathcal{D}|};$$

$w_3(X, \mathcal{D})$ is the sum of the normalized distances from the transactions containing X in the second half of $\mathcal{D}$ to the middle position of $\mathcal{D}$, which is defined as:

$$w_3(X, \mathcal{D}) = \frac{\sum_{\mathcal{T} \in cov(X,\mathcal{D})}\{\rho(\mathcal{T}, \mathcal{D} - \lceil \frac{|\mathcal{D}|}{2} \rceil | \rho(\mathcal{T}, \mathcal{D}) > \frac{|\mathcal{D}|+1}{2}\}}{|\mathcal{D}|};$$

and $w_4(X, \mathcal{D})$ is the sum of the normalized distances from the transactions containing X in the second half of $\mathcal{D}$ to the end of $\mathcal{D}$, which is defined as:

$$w_4(X, \mathcal{D}) = \frac{\sum_{\mathcal{T} \in cov(X,\mathcal{D})}\{|\mathcal{D}| + 1 - \rho(\mathcal{T}, \mathcal{D}) | \rho(\mathcal{T}, \mathcal{D}) > \frac{|\mathcal{D}|+1}{2}\}}{|\mathcal{D}|}.$$

In $w_2(X, \mathcal{D})$ and $w_3(X, \mathcal{D})$, $\lceil x \rceil$ rounds x to the smallest integer that is greater than x.

For example, the weight of pattern G_1G_2 in the *normal* database in Table 1 is $\langle 2.25, 1.6, 0.45, 1.2\rangle$. It tells that in the *normal* database the frequency of G_1G_2 is low at the beginning of the database, but starts to increase in the first half of the database, and achieves its highest value before the middle point of the database, but it becomes lower and lower in the second half of the database. Thus, a weight vector captures the general frequency-changing directions of a pattern in a database.

2.2 Definition of diverging ratio

We are interested in finding patterns whose frequency changes significantly differently in two databases. For this purpose, we need to measure the difference between the weight of a pattern in one database and its weight in the other.

Based on the cosine measure, we define a measure of distance between two weight vectors as follows.

Definition 3. The *diverging ratio* of a pattern X between datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, denoted as $div(X)$, is defined as:[1]

$$div(X) := 1 - \frac{\overrightarrow{\omega}(X, \mathcal{D}_1) \bullet \overrightarrow{\omega}(X, \mathcal{D}_2)}{\|\overrightarrow{\omega}(X, \mathcal{D}_1)\| \times \|\overrightarrow{\omega}(X, \mathcal{D}_2)\|}. \quad (1)$$

where $u \bullet v$ is the *dot product* of two vectors u and v, and $\|v\|$ is the *magnitude* of the vector v.

It's easy to see that the diverging ratio is between 0 and 1. The higher the diverging ratio, the bigger the angle between $\overrightarrow{\omega}(X, \mathcal{D}_1)$ and $\overrightarrow{\omega}(X, \mathcal{D}_2)$, and thus the greater the frequency-change difference of X between the two datasets.

2.3 Definition of diverging pattern

We are interested in finding patterns whose diverging ratio is large, which is defined as follows.

Definition 4. A pattern X is a *Diverging Pattern* (*DP*) with respect to a pair of transaction databases $\mathcal{D}_1$ and $\mathcal{D}_2$, if the following conditions hold:

(1) $sup(X, \mathcal{D}_1) \geq t_s$ and $sup(X, \mathcal{D}_2) \geq t_s$;

(2) $div(X) \geq t_d$;

where t_s and t_d are called *pattern support threshold* and *diverging ratio threshold*, respectively.

3. MINING DIVERGING PATTERNS

In this section, we present an algorithm, called *DP-mine*, for mining the set of diverging patterns from a pair of datasets.

1: Call *FP-weight*$(\mathcal{D}_1, t_s)$ to compute (1) the set $\mathcal{F}_1$ of frequent patterns from $\mathcal{D}_1$ with $min\_sup = t_s$ and (2) for each f_i in $\mathcal{F}_1$, $w_1(f_i, \mathcal{D}_1)$, $w_4(f_i, \mathcal{D}_1)$, $c(f_i, \mathcal{D}_1)$ and $c_1(f_i, \mathcal{D}_1)$, where $c(f_i, \mathcal{D}_1)$ and $c_1(f_i, \mathcal{D}_1)$ are the support counts of f_i in $\mathcal{D}_1$ and the first half of $\mathcal{D}_1$ respectively.

[1]We omit $\mathcal{D}_1$ and $\mathcal{D}_2$ whenever it is clear from the context.

2: Call *FP-weight*$(\mathcal{D}_2, t_s)$ to compute (1) the set $\mathcal{F}_2$ of frequent patterns from $\mathcal{D}_2$ with $min\_sup = t_s$ and (2) for each f_i in $\mathcal{F}_2$, $w_1(f_i, \mathcal{D}_2)$, $w_4(f_i, \mathcal{D}_2)$, $c(f_i, \mathcal{D}_2)$ and $c_1(f_i, \mathcal{D}_2)$, where $c(f_i, \mathcal{D}_2)$ and $c_1(f_i, \mathcal{D}_2)$ are the support counts of f_i in $\mathcal{D}_2$ and the first half of $\mathcal{D}_2$ respectively.
3: $\mathcal{S} \leftarrow \mathcal{F}_1 \bigcap \mathcal{F}_2$
4: $\mathcal{S}_{DP} \leftarrow \emptyset$
5: **for all** $P \in \mathcal{S}$ **do**
6: Compute $w_2(P, \mathcal{D}_1)$ and $w_3(P, \mathcal{D}_1)$;
7: Compute $w_2(P, \mathcal{D}_2)$ and $w_3(P, \mathcal{D}_2)$;
8: Compute $div(P)$;
9: **if** $div(P) \geq t_d$ **then**
10: $\mathcal{S}_{DP} \leftarrow \mathcal{S}_{DP} \bigcup \{P\}$
11: **end if**
12: **end for**
13: **return** $\mathcal{S}_{DP}$

There are two major phases in this algorithm. During the first phase (Step 1 and Step 2), all frequent itemsets in the two datasets along with their support counts, support counts in the first half of the databases and two of their weight components are derived using *FP-weight* (to be discussed below), with t_s as the minimum support threshold. In the second phase (starting from Step 3 to the end), the algorithm finds all the diverging patterns between the datasets based on the set of frequent itemsets. In Step 3, patterns that are frequent in both $\mathcal{D}_1$ and $\mathcal{D}_2$ are collected in set $\mathcal{S}$. From step 5 to 10, for each pattern in $\mathcal{S}$ the algorithm first computes the two other weights that were not calculated in *FP-weight* based on the weights and support counts collected in *FP-weight*. Then it computes the diverging ratio of the pattern based on the pattern's weights in the two datasets. If the pattern's diverging ratio is greater than or equal to t_d, the pattern is a diverging pattern and is added into the set $\mathcal{S}_{DP}$, which is the output of the algorithm.

The computation of the two remaining weight components in Steps 6 and 7 is done as follows:

$$w_2(X, \mathcal{D}) = c_1(X, \mathcal{D}) \times \lceil \frac{|\mathcal{D}| + 1}{2} \rceil - w_1(X, \mathcal{D}), \text{ and}$$

$$w_3(X, \mathcal{D}) = (c(X, \mathcal{D}) - c_1(X, \mathcal{D})) \times \lceil \frac{|\mathcal{D}| + 1}{2} \rceil - w_4(X, \mathcal{D}).$$

where $c_1(X, \mathcal{D})$, $c(X, \mathcal{D})$, $w_1(X, \mathcal{D})$ and $w_4(X, \mathcal{D})$ were calculated by *FP-weight* in Steps 1 and 2. It is trivial to prove that the above relations hold according to Definition 2.

The *FP-weight* function extends the *FP-growth* algorithm [8] to find all the frequent patterns from a dataset and at the same time to compute two of the weight components of each frequent pattern together with the pattern's counts in the whole and first half of the database. The *FP-growth* algorithm first constructs a FP-tree, which maps each transaction into a path of the tree, while keeping only the frequent items. Each tree node is associated with the count of transactions that contain the items on the path from the root to the node. To incorporate weight computation into the mining process, we also associate each node with three other fields: w_1, w_4 and c_1, where w_1 holds the sum of the distances between the beginning of the dataset and the transactions in the first half of the database which contain the items on the path from the root to the node, c_1 holds the number of such transactions, and w_4 holds the sum of the distances between the end of the dataset and the transactions in the second half of the dataset containing the items on the path from the root to the node. During the FP-tree building process, when processing a transaction $\mathcal{T}$, if the count value of a node is incremented by 1, we also add $\rho(\mathcal{T}, \mathcal{D})$ to the w_1 field of the node and increment the count in the c_1 field if $\mathcal{T}$ is in the first half of the data set (i.e., $\rho(\mathcal{T}, \mathcal{D}) < \frac{|\mathcal{D}+1|}{2}$); if $\rho(\mathcal{T}, \mathcal{D}) > \frac{|\mathcal{D}+1|}{2}$), $|D| + 1 - \rho(\mathcal{T}, \mathcal{D})$ is added to the w_4 field of the node. Thus, after the tree is built, the w_1, w_4 and c_1 fields hold their corresponding values for the the whole dataset. During the recursive mining process of *FP-growth*, whenever there is a need to compute the count of a pattern in a conditional pattern base, w_1, w_4 and c_1 of the pattern are computed in the same way.

Table 2: Characteristics of data sets

data set	# of items	# of trans	# of *FPs*	# of joined FPs	# of *DPs*
A_1	497	18451	906	555	28
A_2		41151	854		
B_1	3340	44760	9235	1322	26
B_2		32752	4794		
C_1	1657	223957	27791	24414	1874
C_2		291640	27957		
M_1	294	19096	1731	1554	0
M_2		13615	1837		
R_1	16470	36251	3032	1882	80
R_2		51911	2803		

$t_s = 0.2\%$ $t_d = 0.3$
A = *BMS-WebView-1* M = *MSweb*
B = *BMS-WebView-2* R = *Retail*
C = *BMS-POS*

4. EXPERIMENTAL EVALUATION

We conducted experiments on several real-world data sets with different characteristics, as shown in Table 2. The first three data sets[2] were contributed by Blue Martini Software as the KDD Cup 2000 data [15]. *BMS-WebView-1* and *BMS-WebView-2* contain several months worth of clickstream data from two e-commerce web sites. *BMS-POS* contains several years worth of point-of-sale data from a large electronics retailer. Data in *MSweb*[3] was obtained from UCI Machine Learning Repository. It records the use of www.microsoft.com by 38000 anonymous, randomly-selected users. For each user, the data lists all the areas of the web site that the user visited in one week. *Retail*[4] is taken from the Frequent Itemset Mining Implementations Repository. It contains the (anonymized) retail market basket data from an anonymous Belgian retail store [5]. For each data set, we randomly select a split point between the 30th and 70th percentiles of the data to partition the data set into two disjoined subsets. For example, *BMS-WebView-1* is partitioned into A_1 and A_2, where A_1 contains the first 18,451 transactions and A_2 contains the last 41,151 transactions.

The last three columns of Table 2 list the number of frequent patterns in each subset, the number of patterns that are frequent in both subsets, and the number of diverging patterns generated from each pair of data sets, with $t_s = 0.2\%$ and $t_d = 0.3$, respectively. As can be seen, when the pattern support threshold is low, a huge number of frequent patterns can be generated from these data sets. In contrast, the number of diverging patterns discovered by our approach is generally much smaller. Note that the number of diverging patterns discovered from a pair of data sets depends on the underlying distributions of patterns in the two data sets. Our approach enables us to identify those frequent patterns that are actually different between the two data sets but cannot be distinguished by frequent pattern mining.

[2]http://www.ecn.purdue.edu/KDDCUP/data/
[3]http://kdd.ics.uci.edu/databases/msweb/msweb.html
[4]http://fimi.cs.helsinki.fi/data/retail.dat

Table 3: Selected diverging patterns

Pattern (X)	$sup(X, \mathcal{D}_1)$ (%)	$sup(X, \mathcal{D}_2)$ (%)	$\overrightarrow{\omega}(X, \mathcal{D}_1)$	$\overrightarrow{\omega}(X, \mathcal{D}_2)$	$div(X)$ (%)
A\{309, 314\}	2.56	1.78	$\langle 40.9, 10.6, 110.1, 74.9\rangle$	$\langle 127, 0, 204.5, 7.1, 26.9\rangle$	70.0
B\{429, 1130\}	0.25	0.36	$\langle 2.0, 0.5, 38.8, 13.7\rangle$	$\langle 13.6, 29.9, 8.5, 7.5\rangle$	66.7
C\{880, 1189\}	0.21	0.31	$\langle 82.0, 96.5, 24.0, 32.5\rangle$	$\langle 53.8, 26.7, 182.5, 183.0\rangle$	50.9
R\{39, 48, 389\}	0.38	0.35	$\langle 5.4, 5.1, 49.4, 9.6\rangle$	$\langle 20.0, 40.0, 15.1, 16.9\rangle$	52.2

We conducted the K-S test [13] on all the diverging patterns discovered from the above five databases with $t_s = 0.2\%$ and $t_d = 0.3$. The results show that the distributions of each discovered diverging pattern in the two corresponding datasets are significantly different with p-values less than 1%. Table 3 describes some discovered diverging patterns (one for each dataset with discovered diverging patterns) in terms of their support values (column 2 and 3), weights (column 4 and 5), and diverging ratio (column 6).

We also conducted the experiments with other support and diverging ratio thresholds. As expected, at any fixed diverging ratio threshold (or pattern support threshold), the number of generated diverging patterns decreases with the increase of the pattern support threshold (or diverging ratio threshold). We also tested the scalability of *DP-mine*. Our result indicates that *DP-mine* has linear scalability against the number of transactions. Detailed results are omitted due to the space limit.

5. RELATED WORK

Discovery of useful distinguishing features between data sets is an important objective in data mining. The concept of Emerging Patterns was first introduced in [6] to capture changes or differences between data sets. An Emerging Pattern (EP) is defined as an itemset X satisfying $growthrate(X) = \frac{sup(X,D_2)}{sup(X,D_1)} \geq g$, where D_1, D_2 are two different data sets and $g > 1$ is called the growth rate threshold. Several variants of emerging patterns have been introduced, with Jumping Emerging Patterns (JEPs) being the most important one [10]. JEPs are emerging patterns with infinity growth rate. Since a JEP can only be found in one distinct class in the database, it is a good indicator of that class. Other variants of emerging patterns include strong emerging patterns (which are emerging patterns with all subsets being emerging patterns) and the Most Expressive JEPs (which are the minimal JEPs) [10]. Different from diverging patterns, emerging patterns do not consider how frequency of a pattern changes within a dataset and how this change differs from the change in another dataset.

The problem of contrast-set mining was introduced in [3, 4], and the STUCCO algorithm was proposed to efficiently search through the space of contrast-sets. Contrast-sets are defined as conjunctions of attributes-value pairs that differ meaningfully in their probabilities across several groups. They can be used to identify differences between groups. Follow-up work in [14] discovered that existing commercial rule-finding system, Magnum Opus, can successfully perform the contrast-set task. The authors concluded that contrast-set mining is a special case of the more general rule-discovery task. None of the above work addresses the problem of finding patterns whose frequency changes in different directions in two contrast data sets.

6. CONCLUSIONS

We have defined a new type of contrast patterns, called *diverging patterns*, to represent the patterns whose frequency changes significantly differently in two contrast data sets. We use a four-dimensional vector to represent each pattern, and define the pattern's diverging ratio based on the angular difference between its vectors in two datasets. Furthermore, an algorithm called *DP-mine* is developed, which makes use of a standard frequent pattern mining algorithm to compute relevant vectors efficiently. Finally, we present experimental results on real-world data sets, showing that *DP-mine* can effectively and efficiently reveal new and interesting knowledge from large databases.

7. REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of ACM SIGMOD'93*, pages 207–216, 1993.

[2] J. Bailey, T. Manoukian, and K. Ramamohanarao. Fast algorithms for mining emerging patterns. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 39–50, 2002.

[3] S. D. Bay and M. J. Pazzani. Detecting change in categorical data: mining contrast sets. In *Proc. of the 5th ACM SIGKDD*, pages 302–306, 1999.

[4] S. D. Bay and M. J. Pazzani. Detecting group differences: mining contrast sets. *Data Mining Know. Discov.*, 5(3):213–246, 2001.

[5] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets. Using association rules for product assortment decisions: a case study. In *Proc. of the 5th ACM SIGKDD*, pages 254–260, 1999.

[6] G. Dong and J. Li. Efficient mining of emerging patterns: discovering trends and differences. In *Proc. of the 5th ACM SIGKDD*, pages 43–52, 1999.

[7] H. Fan and K. Ramamohanarao. Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers. *IEEE TKDE*, 18(6):721–737, 2006.

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

[9] S. Kullback. Information Theory and Statistics, Dover Publications Inc., Mineola, New York, 1968.

[10] J. Li, G. Dong, and K. Ramamohanarao. Making use of the most expressive jumping emerging patterns for classification. In *Proc. of the 4th PAKDD*, pages 220–232, 2000.

[11] B. Liu, W. Hsu, and Y. Ma. Discovering the set of fundamental rule changes. In *Proc. of the 7th ACM SIGKDD*, pages 335–340, 2001.

[12] E. Loekito and J. Bailey. Fast mining of high dimensional expressive contrast patterns using zero-suppressed binary decision diagrams. In *Proc. of the 12th ACM SIGKDD*, Philadelphia, USA, 2006.

[13] A. Stuart, K. Ord, and S. Arnold. *Kendall's Advanced Theory of Statistics*. 1999.

[14] G. I. Webb, S. Butler, and D. Newlands. On detecting differences between groups. In *Proc. of the 9th ACM SIGKDD*, pages 256–265, 2003.

[15] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In F. Provost and R. Srikant, editors, *Proc. of the 7th ACM SIGKDD*, pages 401–406, 2001.