Automatic Detection Of Vaccine Adverse Reactions By Incorporating Historical Medical Conditions

Zhonghua Jiang
Computer Science & Engineering
University of Minnesota, Twin Cities
zjiang@cs.umn.edu

George Karypis
Computer Science & Engineering
University of Minnesota, Twin Cities
karypis@cs.umn.edu

Abstract

This paper extends the state of art by bringing the historical medical conditions into the vaccine adverse reaction discovery process. The goal is to identify evidences which suggest that given adverse reaction is likely to be developed for individuals with certain medical conditions when they are vaccinated with certain vaccines. We propose a novel measure called *dual-lift* for this task. It is shown that the *dual-lift* measure can not only identify medical conditions associated with known vaccine adverse reactions, but also have the potential of detecting new adverse reactions that are otherwise hidden. We formulate this problem in the framework of constraint pattern mining. Three constraints are developed. The first is based on the *dual-lift* measure which aims to discover meaningful patterns, the second is used to remove redundancy, and the third is used to ensure prevalence of generated patterns. We present a pattern mining algorithm *DLiftMiner* which utilizes a novel approach to upper bound the *dual-lift* measure for reducing the search space. Experimental results show that our pruning methods lead to dramatic performance improvement. It is also shown that *DLiftMiner* scales linearly with the size of input database. Some interesting vaccine adverse reactions discovered from VAERS database are presented.

1. INTRODUCTION

Detecting adverse reactions of marketed vaccines is a well known important but challenging task. Due to statistical and clinical limitations, preapproval trials may fail to identify vaccine adverse reactions. This is where the postmarketing surveillance of spontaneous reporting system takes on special importance. The Vaccine Adverse Event Reporting System (VAERS) is such a system co-managed by the Food and Drug Administration (FDA) and the Centers for Disease Control and Prevention (CDC) [1]. It collects information about adverse reactions (possible side effects) that occur after the administration of vaccines licensed for use in the United States.

Existing approaches for adverse reaction detection ([2], [3], [4], [5], [6]) focus on vaccine-adverse reaction pairs that are statistically overrepresented and deserve further investigations. Potentially rich background information associated with the reports are ignored. In fact, in VAERS, each report contains historical medical conditions under which the vaccine adverse reactions are developed, such as, medications the vaccine recipient was taking, pre-existing physician diagnosed allergies and birth defects, and any illness at the time of vaccination. This information may potentially be very informative, as it will allow us to identify sub-groups of individuals (with specific medical conditions) that are more susceptible to given vaccine adverse reactions.

The focus of this paper is to develop efficient methods to identify patterns of the form $\langle$*medical conditions*, *vaccines*, *adverse reactions*$\rangle$, which suggests that the *adverse reactions* are likely to be developed for individuals with *medical conditions* when they are vaccinated with *vaccines*. It makes three key contributions. The first contribution is that it introduces a novel measure called *dual-lift* to capture correlations between *vaccines*, *adverse reactions* and *medical conditions*. The second contribution is that it formulates the problem of discovering vaccine adverse reactions with historical medical conditions as a constraint pattern mining problem. Three constraints are developed. The first is based on the *dual-lift* measure which aims to discover meaningful patterns, the second is used to remove redundancy, and the third is used to ensure prevalence of generated patterns. The third contribution is that it develops a computationally efficient algorithm called *DLiftMiner* (which stands for *Dual-Lift Miner*) for mining constraint patterns in VAERS database, that utilizes a novel approach to upper bound the *dual-lift* measure for reducing the search space.

The *DLiftMiner* algorithm is evaluated on a series of synthetically generated datasets. It is shown that the performance is improved dramatically due to the effective pruning of the search space. It is also shown that *DLiftMiner* scales linearly with the size of input database. Furthermore we evaluate the ability of the *dual-lift* measure and the *DLiftMiner* algorithm to find interesting vaccine adverse reactions on VAERS database. It is shown that the *dual-lift* measure can not only find medical conditions associated with known vaccine adverse reaction pairs, but also have the potential of detecting new adverse reactions that are otherwise hidden.

The remaining of this paper is organized as follows. Section 2 presents related works to constraint pattern mining and adverse reaction detection. Section 3 introduces notations and definitions that are used through out the rest of the paper. Section 4, presents the *dual-lift* measure. Section 5 formulates the constraint pattern mining problem. The *DLiftMiner* algorithm is explained in Section 6. Experimental results with both synthetic and real world datasets are included in Section 7. Finally Section 8 provides some concluding remarks.

2. RELATED WORK

Among the vast amount of work related to the problem of pattern mining, the research that is directly relevant to this work are the approaches that have been developed for dealing with tough constraints [7]. A constraint is tough if it is neither anti-monotone nor monotone, and cannot be converted to either anti-monotone or monotone constraint [7]. Implementing effective pruning strategies is crucial for pushing tough constraints into pattern mining process. For example, a new class of tough constraint called *Loose Anti-Monotone* constraint is introduced in [8]. And the pruning strategy in [8] is based on the fact that a transaction can be deleted if it is not superset of at least one frequent k-itemset satisfying the constraint at any iteration $k \geq 2$. Another example is that LPMiner [9] utilizes smallest valid extension (SVE) property for pruning when dealing with the length decreasing support constraint.

The approach we take in this work is based on the idea of boundable constraint [10]. There are a few other works which explore this same idea. For instance, in [11], an upper bound of chi-square is derived based on convexity of chi-square function. And in [12], the upper bound of Pearson's correlation is shown to have conditional monotone property. In [13], the pruning strategy is to upper bound the *support*, $confidence$ and a new measure called *improvement*.

Another category of related works has to do with adverse reaction discovery. Different measures were proposed for this task, such as, the proportional reporting ratio (PRR) [2], the empirical Bayes geometric mean (EBGM) [3], and the reporting odds ratio (ROR) [4]-[5]. Bate *et al* [6] uses information component (IC) as the measure of associations, which is most close to our work since the information component is defined as the logarithm of *lift*. However, to the best of our knowledge, no prior work has taken into account historical medical conditions explicitly into the adverse reaction discovery problem. And our measure *dual-lift* is proposed specifically for this task.

3. DEFINITIONS AND NOTATION

A *vaccine-adverse reaction* database DB is a set of records of the form $\langle tid, M, V, R\rangle$, where *tid* is a unique record id. Each record can be interpreted as that an individual with historical medical conditions M developed adverse reactions R after the vaccination of V. We will refer to the sets of items in M, V, and R as the m-items, v-items and r-items, respectively.

A record $\langle tid, M, V, R\rangle$ is said to contain itemset I if $I \subseteq M \cup V \cup R$. The *support* of I in DB, denoted by $supp(I|DB)$, is the total number of records in DB that contain I. The *transaction count* of I in DB, denoted by $tcnt(I|DB)$, is the number of records for which $I = M \cup V \cup R$. An association rule is of the form $I \rightarrow r$, where I is the prefix itemset and r is the suffix itemset. The *confidence* of $I \rightarrow r$ is defined as

$$conf(I \rightarrow r|DB) = \frac{supp(I \cup r|DB)}{supp(I|DB)}. \quad (1)$$

The *lift* of $I \rightarrow r$ is defined as

$$lift(I \rightarrow r|DB) = \frac{conf(I \rightarrow r|DB)}{conf(\emptyset \rightarrow r|DB)}, \quad (2)$$

where $\emptyset$ is the null set.

A *conditional database* $DB_{|A}$ is a set of records from DB which contain itemset A. A *projected database* DB_B is formed by projecting all records from DB onto itemset B, that is, removing all items which are not in B from each record in DB. A *projected conditional database* $DB_{B|A}$ is defined as projected database from $DB_{|A}$ on itemset B. Let set $DB^{(d)}$ contain all itemsets whose transaction counts in DB are non-zero, and similarly $DB_B^{(d)}$ ($DB_{B|A}^{(d)}$) contain all itemsets whose transaction counts in DB_B ($DB_{B|A}$) are non-zero.

4. DUAL-LIFT MEASURE

Our objective is to identify all patterns of the form $\langle M, V, R\rangle$ such that there are evidences to suggest adverse reactions R are likely to be developed if individuals with historical medical conditions M are vaccinated with V. The key step towards this goal is to develop a measure to capture correlations among M, V, and R.

One may consider to use $lift(M \cup V \rightarrow R|DB)$ as the measure. However, this measure has the weakness that M and V are not distinguished from each other, so that the correlation may be introduced by one of them, say V, but not the other one, say M. Another possibility is

$$\mathcal{L} = \frac{P(M \cup V \cup R|DB)}{P(M|DB)P(V|DB)P(R|DB)}, \quad (3)$$

where $P(I|DB)$ (I is M, V, R or $M \cup V \cup R$) is the probability of I estimated in DB (for example, $P(I|DB)$ can be estimated by $conf(\emptyset \rightarrow I|DB)$). The problem with this measure is that $\mathcal{L}$ can be high if any two of the three sets are highly correlated. In fact, it is easy to see that $\mathcal{L} = lift(M \cup V \rightarrow R|DB)lift(M \rightarrow V|DB)$. So $\mathcal{L}$ can be high if only M and V are highly correlated.

Instead, we propose to use the following measure:

DEFINITION 4.1 (*dual-lift*). *The dual-lift of a pattern* $\langle M, V, R\rangle$, *denoted by* $dual\_lift(\langle M, V, R\rangle|DB)$, *is defined as:*

$$\min(lift(V \rightarrow R|DB_{|M}), lift(M \rightarrow R|DB_{|V})). \quad (4)$$

The *dual-lift* measure combines two quantities. The first, $lift(V \rightarrow R|DB_{|M})$, is designed to detects vaccine adverse reaction pairs $\langle V, R\rangle$ in a subset of records containing medical conditions M. This is because if individuals with medical conditions M are susceptible to $\langle V, R\rangle$, we should be able to detect $\langle V, R\rangle$ in $DB_{|M}$. Another advantage of this *lift* measure is that we can uncover potentially important vaccine

adverse reaction pairs $\langle V, R\rangle$ ($lift(V \rightarrow R|DB_{|M})$ is high), which are difficult to detect by looking at the whole database ($lift(V \rightarrow R|DB)$ is low). The second, $lift(M \rightarrow R|DB_{|V})$, is designed to identify individuals with medical conditions M which have higher chance of developing adverse reactions R when vaccinated with V. This is because $lift(M \rightarrow R|DB_{|V})$ is high implies that $conf(M \cup V \rightarrow R|DB)$ is larger than $conf(V \rightarrow R|DB)$. However, there is a limitation when these two quantities are used alone, that is, a high value of $lift(V \rightarrow R|DB_{|M})$ does not imply that M is correlated with $\langle V, R\rangle$, and similarly a high value of $lift(M \rightarrow R|DB_{|V})$ does not imply that V is correlated with $\langle M, R\rangle$. The *dual-lift* measure is designed to overcome this limitation by combining them together.

5. PROBLEM FORMULATION

In this section, we formulate the problem of detecting vaccine adverse reactions with historical medical conditions in the framework of constraint pattern mining. In our formulation, there are three components that define the interesting patterns: (i) minimum *dual-lift* constraint, which utilizes the *dual-lift* measure developed in Section 4, (ii) minimum *improvement* constraint, which aims to remove *redundant* patterns, and (iii) minimum *support* constraint, which defines the set of *frequent* patterns. The formal definitions of these components are as follows:

DEFINITION 5.1 (MINIMUM *dual-lift* CONSTRAINT). *The pattern $\langle M, V, R\rangle$ satisfies the minimum dual-lift constraint of $l_0 > 0$ in database DB if*

$$dual\_lift(\langle M, V, R\rangle|DB) \geq l_0. \tag{5}$$

DEFINITION 5.2 (MINIMUM *improvement* CONSTRAINT). *The pattern $\langle M, V, R\rangle$ satisfies the minimum improvement constraint of $m_0 \geq 0$ if*

$$dual\_lift(\langle M, V, R\rangle|DB) - dual\_lift(\langle M', V', R'\rangle|DB) \geq m_0, \tag{6}$$

for any of its proper sub-pattern $\langle M', V', R'\rangle$ (that is, $M' \subseteq M \wedge V' \subseteq V \wedge R' \subseteq R$ and $M' \cup V' \cup R' \neq M \cup V \cup R$), satisfying the minimum dual-lift constraint of l_0. A pattern which does not satisfy minimum improvement constraint is said to be redundant.

DEFINITION 5.3 (MINIMUM *support* CONSTRAINT). *The pattern $\langle M, V, R\rangle$ satisfies minimum support constraint of $s_0 > 0$ if*

$$supp(M \cup V \cup R|DB) \geq s_0. \tag{7}$$

A pattern satisfying minimum support constraint is called a frequent pattern.

The idea of introducing the minimum *improvement* constraint is borrowed from [13]. The goal is to select a subset of good patterns which can be presented to domain experts. Intuitively, Definition 5.2 says that a pattern should be considered redundant if it does not achieve sufficient improvement over its proper sub-patterns satisfying minimum *dual-lift* constraint. This is because the correlation observed in this pattern may have been captured in its sub-patterns already.

In this paper, we only consider the case when R contains a single reaction. With the aid of above definitions, our problem can be stated as:

PROBLEM 5.4. *Given database DB and user specified parameters l_0, m_0 and s_0, identify all possible patterns $\langle M, V, r\rangle$ satisfying minimum dual-lift constraint of l_0, minimum improvement constraint of m_0 (non-redundant) and minimum support constraint of s_0 (frequent), where M is a set of m-items, V is a set of v-items, and r is a set containing a single r-item.*

Algorithm 1 DLiftMiner

Input: An input vaccine adverse reaction database DB^0

1: read records in DB^0; count frequencies of m-items and v-items.
2: read records again and construct database DB as a *FP-Tree*, where m-items and v-items are internal nodes and r-items are leaves. m-items and v-items are sorted based on their frequencies from large to small along the tree from top to bottom.
3: remove infrequent items in database DB.
4: Let M_1, V_1, R_1 be sets of m-items, v-items, r-items in DB.
5: $DB^m \leftarrow DB_{M_1 \cup R_1}$; $DB^v \leftarrow DB_{V_1 \cup R_1}$
6: **for** r in R_1 **do**
7: $\quad l[r] \leftarrow l_0$
8: **end for**
9: $M_0 \leftarrow \emptyset$; $V_0 \leftarrow \emptyset$
10: branch_and_bound(DB, DB^m, DB^v, l, M_0, V_0)

6. DLIFTMINER ALGORITHM

The *DLiftMiner* (see Algorithm 1) algorithm follows the depth-first approach for pattern mining and grows the patterns by adding one item at a time [14]. It reduces the size of the dataset successively by constructing the conditional database which is the subset of transactions that contain the current pattern that is being grown. The database is stored using the FP-tree [15] data structure.

The core of the *DLiftMiner* algorithm is the *branch_and_bound* algorithm that is sketched in Algorithm 2. The *branch_and_bound* algorithm starts with calculating for each r-item r of DB the *dual-lift* for the pattern $\langle M_0, V_0, r\rangle$ (lines 2-6). If the calculated *dual-lift* is larger or equal to its threshold for the corresponding r-item, $l[r]$ is updated (line 9). Then it determines whether the next item i should be conditional on by finding the set R which contains frequent r-items in DB' (or $DB_{|i}$) whose *dual-lift* upper bound is larger than or equal to current threshold. The *dual-lift* upper bounds are calculated in lines 24-25 as *dual-lift-bnd*$_r$, where bnd^m_r (bnd^v_r) is the upper bound of $lift(V'_0 \cup V \rightarrow r|DB^0_{|M'_0 \cup M})$ ($lift(M'_0 \cup M \rightarrow r|DB^0_{|V'_0 \cup V})$) for any frequent patterns $\langle M, V, r\rangle$ in DB'. If R is not empty, new databases DB', DB'^m, DB'^v are constructed with necessary items removed (lines 30-36), and *branch_and_bound* is called with updated parameters (line 35). Note that there is no need to actually construct DB',

Algorithm 2 branch_and_bound(DB, DB^m, DB^v, l, M_0, V_0)

Input: l contains the current *dual-lift* threshold for each r-item. M_0 and V_0 are the sets of m-items and v-items that have been searched. DB is $DB^0_{|M_0 \cup V_0}$ with items in $M_0 \cup V_0$ and other (for example, infrequent) items removed. DB^m is the projected database from $DB^0_{|M_0}$ onto m-items and r-items of DB. DB^v is the projected database from $DB^0_{|V_0}$ onto v-items and r-items of DB.

1: **for** r in all r-items of DB **do**
2: $S_r \leftarrow supp(r|DB)/|DB|$
3: $S^m_r \leftarrow supp(r|DB^m)/|DB^m|$
4: $S^v_r \leftarrow supp(r|DB^v)/|DB^v|$
5: $L^m_r \leftarrow S_r/S^m_r$, $L^v_r \leftarrow S_r/S^v_r$
6: $dual\text{-}lift_r = \min(L^m_r, L^v_r)$
7: **if** $dual\text{-}lift_r \geq l[r]$ **then**
8: **print** $\langle M_0, V_0, r\rangle$.
9: $l[r] \leftarrow dual\text{-}lift_r + m_0$.
10: **end if**
11: **end for**
12: **while** DB has more m-items or v-items **do**
13: $i \leftarrow$ next m-item or v-item of DB
14: $DB' \leftarrow DB_{|i}$
15: **if** i is m-item **then**
16: $M'_0 \leftarrow M_0 \cup i$, $V'_0 \leftarrow V_0$
17: $DB'^m \leftarrow DB^m_{|i}$, $DB'^v \leftarrow DB^v$
18: **else**
19: $V'_0 \leftarrow V_0 \cup i$, $M'_0 \leftarrow M_0$
20: $DB'^m \leftarrow DB^m$, $DB'^v \leftarrow DB^v_{|i}$
21: **end if**
22: let R be the set of frequent r-items in DB'
23: **for** r in R **do**
24: $bnd^m_r \leftarrow b^u_{r|DB'}/b^l_{r|DB'^m}$, $bnd^v_r \leftarrow b^u_{r|DB'}/b^l_{r|DB'^v}$
25: $dual\text{-}lift\text{-}bnd_r = \min(bnd^m_r, bnd^v_r)$
26: **if** $dual\text{-}lift\text{-}bnd_r < l[r]$ **then**
27: remove r from R
28: **end if**
29: **end for**
30: **if** $|R| > 0$ **then**
31: construct DB'; i and r-items not in R are not copied
32: remove infrequent items in DB'
33: construct DB'^m; remove items that are not in DB'
34: construct DB'^v; remove items that are not in DB'
35: branch_and_bound(DB', DB'^m, DB'^v, l, M'_0, V'_0)
36: **end if**
37: remove i from DB
38: **if** i is m-item **then**
39: remove i from DB^m
40: **else**
41: remove i from DB^v
42: **end if**
43: **end while**

DB'^m and DB'^v in lines 14-20; we can operate on the original databases DB, DB^m and DB^v for evaluating *lift* upper bounds in line 24. Finally, item i is removed (lines 38-42) and the next item is considered (line 13).

Lines 2-6 calculate the *dual-lift* of the current pattern $\langle M_0, V_0, r\rangle$. L^m_r (L^v_r) is the quantity $lift(V_0 \rightarrow |DB^0_{|M_0}|)$ ($lift(M_0 \rightarrow |DB^0_{|V_0}|)$). To see why these calculations are correct, consider:

$$\begin{aligned} & lift(V_0 \rightarrow r|DB^0_{|M_0}) \\ = & \frac{conf(V_0 \rightarrow r|DB^0_{|M_0})}{conf(\emptyset \rightarrow r|DB^0_{|M_0})} \\ = & \frac{conf(\emptyset \rightarrow r|DB^0_{|M_0 \cup V_0})}{conf(\emptyset \rightarrow r|DB^0_{|M_0})} \\ = & \frac{conf(\emptyset \rightarrow r|DB)}{conf(\emptyset \rightarrow r|DB^m)} \\ = & \frac{supp(r|DB)/|DB|}{supp(r|DB^m)/|DB^m|}, \end{aligned} \tag{8}$$

and similarly,

$$lift(M_0 \rightarrow r|DB^0_{|V_0}) = \frac{supp(r|DB)/|DB|}{supp(r|DB^v)/|DB^v|}. \tag{9}$$

6.1 Pruning Strategies

During pattern growth, *DLiftMiner* utilizes various approaches to prune the search space. The support based pruning is used to remove infrequent items. Improvement based pruning is used to set the *dual-lift* thresholds higher and higher. Finally the *dual-lift* based pruning finds the upper bound of the *dual-lift* measure. And if the upper bound is less than current *dual-lift* threshold, the rest of search space is pruned away. Additional details of these pruning methods are provided in the rest of this section.

6.1.1 Support Based Pruning

Support based pruning can be derived from the minimum *support* constraint in Definition 5.3. The idea is to remove items that are considered infrequent. Any r-item is infrequent if its support is less than s_0. In our formulation, every interesting pattern is associated with an r-item. For any m-item or v-item i we define its *maximum support per class* as $\max_r supp(i \cup r|DB)$. Any m-item or v-item whose *maximum support per class* is less than s_0 is considered infrequent and thus removed. Our algorithm applies support based pruning in line 3 of *DLiftMiner* and lines 32-34 of *branch_and_bound*, where infrequent items are removed, and line 22 of *branch_and_bound*, where only frequent r-items are considered.

6.1.2 Improvement Based Pruning

For any r-item r, patterns to be explored by the branch and bound algorithm are super-patterns of $\langle M_0, V_0, r\rangle$. From the definition of minimum *improvement* constraint, the *dual-lift* thresholds for patterns to be explored should be no less than m_0 plus the maximum *dual-lift* values for any proper sub-patterns of $\langle M_0, V_0, r\rangle$ that branch and bound has discovered. These *dual-lift* thresholds are saved in l (line 9). When a new pattern is discovered, its *dual-lift* has to be no less than current *dual-lift* threshold. That means, this new

pattern must have largest *dual-lift* value among its discovered sub-patterns (due to m_0 is non-negative). So updating corresponding *dual-lift* threshold is straightforward: simply add m_0 to *dual-lift* value of the new pattern (line 9).

However, it should be noted that the current implementation of our algorithm does not take into account all the redundancy that can be exploited by Definition 5.2 due to the nature of depth first search. This is because that not all sub-patterns of $\langle M_0, V_0, r\rangle$ were processed before the pattern $\langle M_0, V_0, r\rangle$. For example, assume $M_0 = \{m_1, m_2\}$ and $V_0 = \{v_1\}$ and assume these three items were added to $M_0 \cup V_0$ in the order of m_1, v_1, m_2, then sub-patterns processed before $\langle\{m_1, m_2\}, v_1, r\rangle$ are $\langle\emptyset, \emptyset, r\rangle$, $\langle m_1, \emptyset, r\rangle$, and $\langle m_1, v_1, r\rangle$. Other sub-patterns (for example, $\langle m_2, v_1, r\rangle$) are processed after $\langle\{m_1, m_2\}, v_1, r\rangle$. Extending our current algorithm to properly handle such cases is something that we are currently working on.

6.1.3 Lift Based Pruning

From line 24 in the branch and bound algorithm, we need to find upper bounds for the quantities $lift(V_0' \cup V \to r|DB^0_{|M_0' \cup M})$ and $lift(M_0' \cup M \to r|DB^0_{|V_0' \cup V})$ for any frequent patterns $\langle M, V, r\rangle$ in DB'.

This problem can be tackled by observing that *lift* is the ratio of two *confidences*. In fact,

$$\begin{aligned} & lift(V_0' \cup V \to r|DB^0_{|M_0' \cup M}) \\ = & \frac{conf(V_0' \cup V \to r|DB^0_{|M_0' \cup M})}{conf(\emptyset \to r|DB^0_{|M_0' \cup M})} \\ = & \frac{conf(M \cup V \to r|DB^0_{|M_0' \cup V_0'})}{conf(M \to r|DB^0_{|M_0'})}, \end{aligned} \tag{10}$$

and similarly,

$$lift(M_0' \cup M \to r|DB^0_{|V_0' \cup V}) = \frac{conf(M \cup V \to r|DB^0_{|M_0' \cup V_0'})}{conf(V \to r|DB^0_{|V_0'})}. \tag{11}$$

Because DB' is $DB^0_{|M_0' \cup V_0'}$ with some items removed, DB'^m (DB'^v) is $DB^0_{|M_0'}$ ($DB^0_{|V_0'}$) with some items removed, and $\langle M, V, r\rangle$ is frequent patterns from DB', it is not hard to see that $DB^0_{|M_0' \cup V_0'}$, $DB^0_{|M_0'}$, and $DB^0_{|V_0'}$ can be replaced by DB', DB'^m and DB'^v in equations 10 and 11. So we have

$$lift(V_0' \cup V \to r|DB^0_{|M_0' \cup M}) = \frac{conf(M \cup V \to r|DB')}{conf(M \to r|DB'^m)}, \tag{12}$$

and

$$lift(M_0' \cup M \to r|DB^0_{|V_0' \cup V}) = \frac{conf(M \cup V \to r|DB')}{conf(V \to r|DB'^v)}. \tag{13}$$

Equations 12 and 13 suggest that, to find the upper bound of *lift*, we can find the upper bound of numerator and the lower bound of denominator. So, line 24 is correct if we introduce

- $bnd^u_{r|DB'}$ be upper bound of $conf(M \cup V \to r|DB')$ for any frequent patterns $\langle M, V, r\rangle$ in DB',

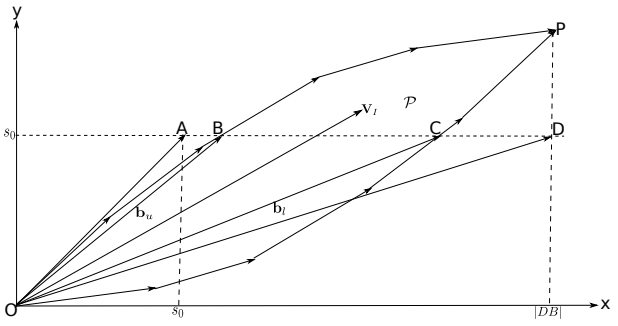

Figure 1: Bound the Confidence

- $bnd^l_{r|DB'^m}$ be lower bound of $conf(M \to r|DB'^m)$ for any frequent patterns $\langle M, \emptyset, r\rangle$ in DB'^m,
- $bnd^l_{r|DB'^v}$ be lower bound of $conf(V \to r|DB'^v)$ for any frequent patterns $\langle \emptyset, V, r\rangle$ in DB'^v.

6.1.4 Bound the Confidence

The above analysis translates the problem of bounding the *lift* into the problem of bounding the *confidence*, which can be defined as:

PROBLEM 6.1 (BOUNDING THE CONFIDENCE). *For the database DB, itemsets I_0, and r, the problem of bounding the* confidence *is to find upper and lower bound for the quantity $conf(I \to r|DB)$ for any $I \subseteq I_0$ such that $supp(I \cup r|DB) \geq s_0$.*

Note that, I_0's for $bnd^u_{r|DB'}$, $bnd^l_{r|DB'^m}$, and $bnd^l_{r|DB'^v}$ discussed above are implicitly defined as all non-r-items of corresponding databases.

There is a simple solution to this problem. The upper bound of $conf(I \to r|DB)$ can be chosen as 1. The lower bound can be derived as

$$conf(I \to r|DB) = \frac{supp(I \cup r|DB)}{supp(I|DB)} \geq s_0/|DB|. \tag{14}$$

However, our goal is to come up with an approach which works significant better than this naive solution.

We introduce the following two lemmas without proof due to limited space [1]:

LEMMA 6.2. *Given database DB, itemsets I_0, and r, for any $I \subseteq I_0$, define vector $\mathbf{V}_I$ whose x-component $V_{I,x}$ is $supp(I|DB_{I_0})$ and y-component $V_{I,y}$ is $supp(I|DB_{I_0|r})$. We have $conf(I \to r|DB)$ is equal to slope of $\mathbf{V}_I$ and $supp(I \cup r|DB)$ is equal to y-component of $\mathbf{V}_I$.*

LEMMA 6.3. *Following the settings in lemma 6.2, for any itemset A, we define vector $\mathbf{v}_A$ whose x-component $v_{A,x}$ is $tcnt(A|DB_{I_0})$ and y-component $v_{A,y}$ is $tcnt(A|DB_{I_0|r})$. Let*

[1]Proofs of lemmas 6.2, 6.3 and theorem 6.4 are included in [16]

Table 1: Parameters for synthetic dataset.

parameter	description	value
ntrans	number of transactions	$10k$
nitems	number of items	500
npats	number of maximal potentially large itemsets	100
patlen	average size of the maximal potentially large itemsets	30
tlen	average size of transactions	30

Table 2: Different versions of *DLiftMiner* algorithms.

algorithm	description
DLiftMiner	applies theorem 6.4 for both upper and lower bounds of confidences
DLiftMiner-U	identical to *DLiftMiner* except confidence upper bound is replaced by 1
DLiftMiner-N	identical to *DLiftMiner* except confidence lower bound is replaced by the naive approach that is, $b^l_{r\|DB} = s_0/\|DB\|$
DLiftMiner-UN	identical to *DLiftMiner* except confidence upper bound is replaced by 1, and confidence lower bound is replaced by the naive approach
DLiftMiner-L	identical to *DLiftMiner* except confidence lower bound is replaced by zero that is, no *lift* pruning is applied

set $\mathcal{A}_I = \{A | A \in DB^{(d)}_{I_0} \wedge I \subseteq A\}$, *then*

$$\mathbf{V}_I = \sum_{A \in \mathcal{A}_I} \mathbf{v}_A. \tag{15}$$

Based on lemmas 6.2 and 6.3, Figure 1 illustrates how we can find an efficient solution to problem 6.1. Let us define $\mathcal{F}$ be the set of vectors $\mathbf{v}_A$ for all A in $DB^{(d)}_{I_0}$. Sorting vectors in $\mathcal{F}$ by slope from large to small and connecting them head to tail gives piece-wise linear curve $\overline{OBP}$. Similarly, sorting vectors in set $\mathcal{F}$ by slope from small to large and connecting them head to tail gives piece-wise linear curve $\overline{OCP}$. If we define another set $\mathcal{P}$ which contains sum of all possible subsets of vectors in $\mathcal{F}$, it can be proved that any vectors in set $\mathcal{P}$ are inside the region enclosed by $\overline{OBP}$ and $\overline{OCP}$. Lemma 6.3 suggests that $\mathbf{V}_I$ is also equal to sum of a subset of vectors from $\mathcal{F}$. So we should have $\mathbf{V}_I \in \mathcal{P}$ and $\mathbf{V}_I$ is inside region $\overline{OBPCO}$. Line $y = s_0$ cuts this region into two parts (we assume that the y-coordinate of point P is larger or equal to s_0). All vectors in $\mathcal{P}$ whose y-component are larger or equal to s_0 are inside the top region $\overline{BCPB}$. From lemma 6.2, any association rule $I \rightarrow r$ $(I \subseteq I_0)$ whose $supp(I \cup r|DB)$ is larger or equal to s_0 is associated with a vector $\mathbf{V}_I$ in the region $\overline{BCPB}$ whose slope is $conf(I \rightarrow r|DB)$. And it can be easily seen that slopes of all vectors inside region $\overline{BCPB}$ are upper bounded by the slope of $\overline{OB}$ and lower bounded by the slope of $\overline{OC}$. This leads to the conclusion that slopes of $\overline{OB}$ and $\overline{OC}$ are upper and lower bounds of $conf(I \rightarrow r|DB)$ (Note that slopes of $\overline{OA}$ and $\overline{OD}$ are upper and lower bounds provided by naive approach discussed before).

This result is summarized into the following theorem:

THEOREM 6.4. *Following the settings in lemma 6.3, define* $\mathcal{F} = \{\mathbf{v}_A | A \in DB^{(d)}_{I_0}\}$ *and sort vectors in* $\mathcal{F}$ *based on their slopes from small to large, and label them as* $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n$, *where* n *is the total number of vectors. Define,*

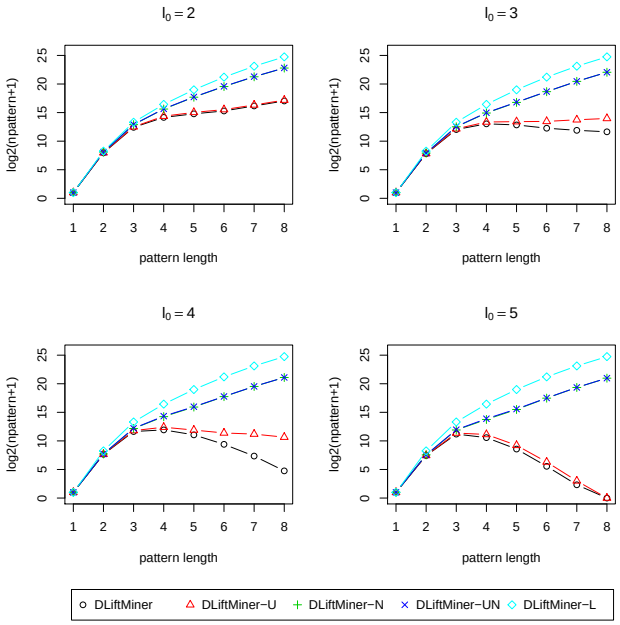

Figure 2: Pruning effects on synthetic dataset

- $b^l_{I_0 \rightarrow r|DB}$ *as slope of the following vector*

$$\mathbf{b}_l = \mathbf{f}_1 + \mathbf{f}_2 + \cdots + \mathbf{f}_{i_l - 1} + \alpha * \mathbf{f}_{i_l}, \tag{16}$$

  where $0 < \alpha \leq 1$, $1 \leq i_l \leq n$ *are chosen such that y-component of* $\mathbf{b}_l$ *is equal to* s_0.
- $b^u_{I_0 \rightarrow r|DB}$ *as slope of the following vector*

$$\mathbf{b}_u = \mathbf{f}_n + \mathbf{f}_{n-1} + \cdots + \mathbf{f}_{i_u + 1} + \beta * \mathbf{f}_{i_u}, \tag{17}$$

  where $0 < \beta \leq 1$, $1 \leq i_u \leq n$ *are chosen such that y-component of* $\mathbf{b}_u$ *is equal to* s_0.

Then we have, $b^l_{I_0 \rightarrow r|DB} \leq conf(I \rightarrow r|DB) \leq b^u_{I_0 \rightarrow r|DB}$, *for any* $I \subseteq I_0$ *such that* $supp(I \cup r|DB) \geq s_0$.

Our implementation of *DLiftMiner* algorithm utilizes a slightly modified *FPTree* data structure. And the bounding approach proposed in theorem 6.4 can be easily applied. This is mainly because each itemset A in $DB^{(d)}_{I_0}$ is mapped to a path along the tree excluding leaf nodes. However, due to limited space, we omit the details but include them into [16].

7. EXPERIMENTAL RESULTS

7.1 Synthetic Datasets

DataSets. To evaluate the performance of our algorithm, we use the IBM Quest market-basket synthetic data generator [17] to generate a series of datasets. The generator takes the parameters described in Table 1. We split the generated items into m-items, v-items and r-items from small id's to large id's so that m-items cover 57%, v-items cover 33% and r-items cover the rest 10%. We keep only the transactions containing all three types of items in the dataset. The parameters for the first synthetic dataset are shown in the

Table 3: Running times for the first synthetic dataset.

l_0 \ algorithm	*DLiftMiner-L*	*DLiftMiner-UN*	*DLiftMiner-N*	*DLiftMiner-U*	*DLiftMiner*
2	87.50	30.93	31.28	2.67	2.63
3	87.50	18.22	18.08	1.15	1.10
4	87.50	9.97	10.10	0.88	0.83
5	87.50	8.73	8.50	0.63	0.63

[a] All times are in minutes

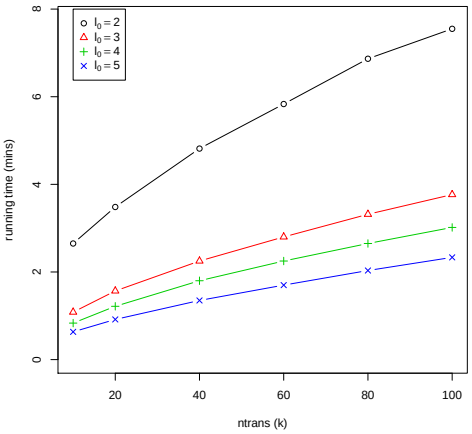

Figure 3: scalability of *DLiftMiner*

Table 4: Number of patterns generated by *DLiftMiner*.

l_0 \ m_0	*NONE*	0	0.5	1.0
2	79355	40517	20558	19171
3	6772	4329	3271	3049

third column of Table 1 and the actual size of the dataset is 7401 records.

In addition, for scalability analysis, we generated another five datasets by setting *ntrans* to be $20k$, $40k$, $60k$, $80k$ and $100k$ (the other parameters are the same as in Table 1). Due to the fact that only transactions with all three types of items are kept, the actual sizes of these datasets are 14824, 29536, 44301, 59036 and 73729.

Experimental Setup. The performance of our algorithm was compared against four successively simpler versions of *DLiftMiner* that incorporate fewer pruning optimizations. These versions are described in Table 2. In our experiment, we set $s_0 = 100$, $m_0 = 0$ and let l_0 take the values 2, 3, 4 and 5. And we limited the maximum length of the patterns discovered to 8 (the length is defined to be the sum of sizes of current sets of m-items and v-items).

Table 5: Fields Extracted from VAERS dataset

VAERS field	description	item type
OTHER_MEDS	narrative about prescription or non-prescription drugs the vaccine recipient was taking at the time of vaccination.	m-item
CUR_ILL	narrative about any illness at the time of vaccination.	
HISTORY	narrative about any pre-existing physician-diagnosed allergies, birth defects, medical conditions that existed at the time of vaccination.	
PRIOR_VAX	prior vaccination event information.	
VAX_TYPE	the set of coded vaccines	v-item
SYMPTOM	the set of MedDRA (Medical Dictionary for Regulatory Activities) coded symptoms	r-item
DIED	died or not	
DISABLE	disability	
ER_VISIT	had ER visit or not	
HOSPITAL	hospitalized or not	
L_THREAT	life threatening or not	
RECOVD	recovered or not	
X_STAY	prolonged hospitalization or not	

Performance of Pruning Methods. The experimental results are summarized in Table 3 and Figure 2. From Table 3, one can see that the pruning methods incorporated into *DLiftMiner* lead to dramatic improvements. Our best algorithm *DLiftMiner* takes only 3.0%, 1.3%, 0.99%, and 0.72% of the running time compared to the baseline approach *DLiftMiner-L* when l_0 takes values 2, 3, 4 and 5, respectively. Even compared with *DLiftMiner-UN*, where only naive pruning is applied, *DLiftMiner* takes only about 7% of its running time.

Table 3 also shows how the different pruning methods impact the performance of *DLiftMiner*. It can be seen that there are dramatic performance improvements when the confidence lower bound is improved (see *DLiftMiner-L* versus *DLiftMiner-UN* and *DLiftMiner-N* versus *DLiftMiner-U*). However the upper bound of *confidence* does not seem to play much role in reducing the running time, so that *DLiftMiner* and *DLiftMiner-U* (*DLiftMiner-N* and *DLiftMiner-UN*) have almost the same results.

Figure 2 plots the number of patterns processed (*npattern*) versus different *pattern length* for different versions of the algorithms. The vertical axis has been scaled to log scale ($\log_2(npattern + 1)$). It can be seen that including *confidence* upper bound does not help much in pruning additional patterns.

Scalability Study. Figure 3 illustrates how the *DLiftMiner* algorithm scales with respect to the size of input databases for four different l_0 values. We set the corresponding mini-

Table 6: Subset of vaccine-symptom associations with medical conditions in S_3.

Symptom	vaccine code(s)	medical condition(s)[a]	dual-lift
Autism	MMR	*HISTORY*-"fever"	4.7
C-reactive protein increased	PNC	*HISTORY*-"Pregnancy"	9.3
Drug toxicity	HEP	*HISTORY*-"Pregnancy"	7.8
Febrile convulsion	DTPHIB	*HISTORY*-"seizure"	3.3
Influenza like illness	LYME	*HISTORY*-"Anxiety"	3.4
Injected limb mobility decreased	PPV	*CUR_ILL*-"allergy"	3.6
Irritability	HIBV	*OTHER_MEDS*-"prevacid"	5.2
Otitis media	(MMR,OPV)	*CUR_ILL*-"Ear infection"	3.0
	(DTP,MMR)		3.4
	(HIBV,MMR,OPV)		3.5
	(DTP,HIBV,MMR)		3.6
	(DTP,MMR,OPV)		3.9
Rash vesicular	VARZOS	*CUR_ILL*-"Hypothyroidism"	3.3
Swelling	(IPV,MMR)	*PRIOR_SYM*-"Swelling" [b]	3.8
Throat tightness	FLU	(*HISTORY*-"Asthma",*HISTORY*-"allergy")	3.7
Urine human chorionic gonadotropin positive	HPV4	*HISTORY*-"Pregnancy"	8.8
Varicella	VARCEL	*CUR_ILL*-"Asthma"	4.2

[a] Medical conditions are expressed in the form of *FIELD_NAME*-"term", where *FIELD_NAME* is from Table 5
[b] *PRIOR_SYM* is symptoms developed from prior vaccination, extracted from *PRIOR_VAX* field.

mum support s_0 to be 1% of the *ntrans* parameter used for generating the dataset. It can be seen that the *DLiftMiner* algorithm scales linearly with the size of input dataset.

Effect of the Minimum Improvement Threshold. To evaluate how different values of m_0 impact the *DLiftMiner* algorithm, we calculated the numbers of patterns generated by *DLiftMiner* for different choices of *improvements* when l_0 takes values 2 and 3. These results are summarized in Table 4, in which, *NONE* means no redundancy is taken into account. One can see that a large fraction of redundant patterns are removed during the mining process when $m_0 = 0$, but setting m_0 higher does not reduce as many patterns. However, we are not able to observe significant additional pruning by incorporating the *improvement* constraint. This may partially due to the fact that our *DLiftMiner* algorithm does not take into account all redundancy defined in Definition 5.2.

7.2 VAERS Dataset

To evaluate the ability of the *dual-lift* measure to identify interesting patterns, we applied the *DLiftMiner* algorithm to the VAERS [1] dataset. At the point of this paper, VAERS contains 21 years (from 1990 to 2010) of vaccination reports plus an additional non-domestic dataset. We downloaded all of them. Fields listed in Table 5 were extracted and transformed to the $\langle M, V, R\rangle$ representation. For example, fields *OTHER_MEDS*, *CUR_ILL*, *HISTORY*, and *PRIOR_VAX* were converted into a set of historical medical conditions (m-items) for each record. The v-items and r-items were constructed similarly.

One challenge we faced was that the fields we used to extract historical medical conditions or m-items were free text and thus not coded. To address this issue, we followed the following process to get a clean set of m-items. First, we split the free text into a bag of words (unigrams). Second, in order to capture some important terms beyond unigrams, we split the text by different delimiters: space, comma, semicolon, period and combinations of them. Third, among the set of all terms generated in the above two steps, we kept only those with frequency greater than or equal to 100. Next, we removed those terms that are obviously non-medical related. Finally, when we observed several terms having the same meanings, we mapped them to a normalized term. For example, "allergy codeine", "allergy to codeine" and "allergic to codeine" were mapped to "codeine allergy". After cleaning up, the total number of terms we got was 1187. For each record and field, the identified terms that were contained in the textual description were included as m-items. For the r-items, we removed those symptoms which do not appear very informative, for example, "Unevaluable event", "Accidental overdose", "Computerised tomogram normal", "Drug exposure during pregnancy" and so on. Also we removed all "injection site" symptoms and all symptoms containing the word "negative".

For our experiment, we set $s_0 = 15$, $l_0 = 3$ and $m_0 = 0$. And we used the *DLiftMiner* algorithm to generate the following sets of patterns:

- S_1 contains all *non-redundant frequent* patterns $\langle V, r\rangle$ such that $lift(V \rightarrow r|DB) \geq {l_0}^2$. S_1 is the solution to the traditional adverse reaction detection problem, where historical medical conditions are not taken into account. We found 3617 patterns in S_1.
- S_2 contains all *non-redundant frequent* patterns $\langle M, V, r\rangle$ satisfying minimum *dual-lift* constraint. S_2 is the solution to Problem 5.4. Experiments show that our idea of combining two *lift* measures with *redundancy*

[2]Since only one *lift* measure is involved, the concept of *redundancy* here should be re-defined accordingly.

Table 7: Subset of vaccine-symptom associations with medical conditions in S_4.

Symptom	vaccine code(s)	medical condition(s)[a]	dual-lift
HOSPITAL-"Y"[b]	HPV4	*OTHER_MEDS*-"estradiol"	3.6
	PNC	*HISTORY*-"diabetes"	3.2
Drug ineffective	RAB	*OTHER_MEDS*-"Hepatitis"	7.5
Erythema	VARCEL	(*CUR_ILL*-"allergy",*HISTORY*-"allergy")	3.6
Herpes zoster	(MMR,VARCEL)	*HISTORY*-"Otitis media"	4.6
Hypotension	TD	*OTHER_MEDS*-"purified protein derivative"	3.3
Joint range of motion decreased	FLU	*HISTORY*-"deficit"	3.8
Lymphadenopathy	MMR	*OTHER_MEDS*-"premarin"	5.5
Nervous system disorder	HEP	*HISTORY*-"Pregnancy"	6.3
Otitis media	(DTP,HIBV,OPV)	*PRIOR_SYM*-"fever"[c]	3.2
Rash	(HIBV,OPV)	*HISTORY*-"penicillin"	3.2
Swelling	IPV	*PRIOR_SYM*-"Swelling"[c]	3.6
	(DTAP,MMR)		3.2
Urticaria	(DTAP,IPV,MMR)	*HISTORY*-"premature"	3.4
Vasodilatation	DTP	(*HISTORY*-"Asthma",*HISTORY*-"allergy")	4.5
	OPV	(*HISTORY*-"allergy",*HISTORY*-"ceclor")	4.2
Wheezing	DTAP	(*HISTORY*-"allergy",*OTHER_MEDS*-"albuterol")	3.0
White blood cell count increased	PPV	*HISTORY*-"Anxiety"	3.8

[a] Medical conditions are expressed in the form of *FIELD_NAME*-"term", where *FIELD_NAME* is from Table 5
[b] *HOSPITAL*-"Y" is extracted from *HOSPITAL* field, meaning the patient is hospitalized.
[c] *PRIOR_SYM* is symptoms developed from prior vaccination, extracted from *PRIOR_VAX* field.

Table 8: Vaccine-Symptom associations from vaccine injury table.

Symptom	Reported vaccine code(s)
Anaphylactic reaction	DT,DTAP,DTAPH,DTP HEP,IPV,MMR,TD
Encephalopathy	DTAP,DTAPH DTP,MMR
Intussusception	RV
Thrombocytopenic purpura	M[a] ,MM[a] ,MMR,MR[a]

[a] Vaccines excluded from analysis due to low or zero frequencies

Table 9: Vaccine-Symptom associations from Vaccine Injury Table found in S_1.

Symptom	Vaccine code(s)	lift
Encephalopathy	(DTAP, HIBV)	3.2
	(HIBV, MMR)	3.1
Intussusception	RV	85.2
	(IPV, RV)	95.5
	(HIBV, RV)	86.4
	(DTAP, RV)	88.2
	(HIBV, IPV, RV)	105.0
	(DTAP, IPV, RV)	101.8
	(DTAP, HIBV, RV)	95.5
	(DTAP, HIBV, IPV, RV)	110.1
Thrombocytopenic purpura	(HIBV , MMR)	4.5

Table 10: Vaccine-Symptom associations from Vaccine Injury Table found in S_2 and S_4

Symptom	Vaccine code(s)	Medical Condition(s)	dual-lift
Anaphylactic reaction	MMR	cur_ill-"allergy"	3.9
	(MMR, VARCEL)	history-"allergy"	3.0

is quite selective. We found 169 patterns in S_2.

- S_3 contains the set of patterns in S_2 whose vaccine reaction pairs $\langle V, r\rangle$ also appear in S_1. S_3 contains the identified medical conditions under which adverse reactions are more likely to be developed for a subset of patterns in S_1. We found 91 patterns S_3.
- S_4 contains the rest of patterns in S_2. S_4 is the set of new vaccine adverse reactions that cannot be discovered without incorporation of historical medical conditions. We found 78 patterns in S_4.

Due to limited space, we include only some of the patterns and the associated *dual-lift* values from S_3 and S_4 into Tables 6 and 7 (see the online supplement [3] for more complete sets of patterns). For each vaccine set V, we present the pattern with the highest *dual-lift*. Note that in Tables 6, 7 (and Tables 9, 10) parenthesis implies vaccine or medical condition interaction. Patterns listed in these two tables can be easily interpreted. For example, the pattern $\langle$*OTHER_MEDS*-"estradiol", HPV4, *HOSPITAL*-"Y"$\rangle$ can be interpreted as that individuals taking the medication estradiol are more likely to be hospitalized when vaccinated with HPV4. However, the actual medical significance of these patterns has to be determined by domain experts.

To evaluate the generated patterns of our approach, we need a list of vaccine adverse reactions (ideally with associated medical conditions) that have been determined significant by domain experts. The Vaccine Injury Table is the only source of this kind that we are aware of [18]-[19] (see Table 8). The Vaccine Injury Table we are using is from [19] with

[3] `www.cs.umn.edu/~zjiang`

coding system for symptoms changing from Coding Symbols for a Thesaurus of Adverse Reaction Terms (COSTART) to MedDRA. Only four of eight symptoms are listed in Table 8 because no patterns were detected for another four (Arthritis, Encephalitis, Brachial plexopathy and Poliomyelitis).

Next, we present the set of patterns in the Vaccine Injury Table that can be discovered by our method. However, be noted that this evaluation is limited due the small number of cases available. Table 9 contains patterns in set S_1 whose symptoms are in Table 8 and vaccine sets have at least one of the reported vaccines for the corresponding symptoms. Our results in Table 9 suggests that vaccine-vaccine interactions play important role when developing these symptoms. Rotavirus vaccine (RV) was licensed on August 31 1998 for routine use in infants in a three-dose series given at two, four and six months of age. In mid-May 1999, nine reports were submitted. On October 14 1999, the vaccine manufacturer voluntarily withdrew its product from the market [20]. Our result in Table 9 shows RV-intussusception can be detected, and it also shows RV can interact with other vaccines and develop intussusception together. Table 10 contains patterns in set S_2 whose symptoms and at least one of the vaccines appear in the Vaccine Injury Table. Only two patterns are detected, both of which also appear in set S_4. This suggests that by incorporating historical medical conditions, we are likely to identify new patterns other than existing well known ones.

8. CONCLUSIONS

In this paper, we formulate the problem of detecting vaccine adverse reactions by incorporating historical medical conditions as a constraint pattern mining problem. We propose to use a novel measure called *dual-lift* to evaluate the significance of patterns. Our pattern mining algorithm *DLift-Miner* utilizes a novel bounding approach for the *dual-lift* measure. Experimental results show that the pruning methods are effective. We also present some interesting vaccine adverse reactions discovered from VAERS database.

Acknowledgment

This work was supported in part by NSF (IIS-0905220, OCI-1048018, and IOS-0820730) and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

9. REFERENCES

[1] C. RT, R. SC, M. JR, and *et al*, "The vaccine adverse event reporting system (vaers)," *Vaccine*, vol. 12, pp. 542–550, 1994.

[2] F. DJ, "Systemic signalling of adverse reactions to drugs," *Methods Inf Med*, vol. 13, pp. 1–10, 1974.

[3] D. W, "Bayesian data mining in large frequency tables, with an application to the fda spontaneous reporting system," *Am Statistician*, vol. 53, pp. 177–190, 1999.

[4] V. P. E, D. W, and van Groothest K, "Application of quantitative signal detection in the dutch spontaneous reporting system for adverse drug reactions," *Drug Saf*, vol. 26 (5), pp. 293–301, 2003.

[5] V. der Heijden P, van Puijenbroek E, and van Buuren S *et al*, "On the assessment of adverse drug reactions from spontaneous reporting systems: the influence of under-reporting on odds ratios," *Stat Med*, vol. 21, pp. 2027–44, 2002.

[6] B. A, L. M, and E. I. *et al*, "A bayesian neural network method for adverse drug reaction signal generation," *Eur J Clin Pharmacol*, vol. 54, pp. 315–21, 1998.

[7] J. Pei and J. Han, "Can we push more constraints into frequent pattern mining?," *In Proceedings of ACM SIGKDD'00*, 2000.

[8] F. Bonchi and C. Lucchese, "Pushing tougher constraints in frequent pattern mining," *In Proc. of PAKDD*, 2005.

[9] M. Seno and G. Karypis, "Lpminer: An algorithm for finding frequent itemsets using length-decreasing support constraint," *ICDM'01, Nov*, 2001.

[10] L. D. Raedt and A. Zimmermann, "Constraint-based pattern set mining," *In SDM. SIAM*, 2007.

[11] S. Morishita and J. Sese, "Traversing itemset lattices with statistical metric pruning," *POD, Dallas, T X USA. ACM*, 2000.

[12] H. Xiong, S. Shekhar, P.-N. Tan, and V. Kumar., "Exploiting a support-based upper bound of pearson's correlation coefficient for efficiently identifying strongly correlated pairs," *KDD'04, August, Seattle, Washington, USA. ACM*, pp. 22–25, 2004.

[13] R. J. B. Jr., R. Agrawal, and D. Gunopulos, "Constraint-based rule mining in large, dense databases," *In Proc. of the 15th Int'l Conf. on Data Engineering*, pp. 188–197, 1999.

[14] M. J. Zaki, "Scalable algorithms for association mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, pp. 372–390, 2000.

[15] J. Han, J. Pei, and Y. Yin, *Mining frequent patterns without candidate generation*, vol. 8, pp. 1–12. ACM Press, 2000.

[16] Z. Jiang and G. Karypis, "Automatic detection of vaccine adverse reactions by incorporating historical medical conditions," Tech. Rep. 11-007, Department of Computer Science and Engineering, University of Minnesota, 2011.

[17] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," *Proceedings of the 20th VLDB Conference, Santiago, Chile*, 1994.

[18] S. KR, H. CJ, and J. R. (eds), *Adverse Events Associated with Childhood Vaccines: Evidence Bearing on Casuality*. Vaccine Safety Committee, Institute of Medicine: National Academy Press: Washington, DC, 1994.

[19] D. Banks, E. J. Woo, and *et al*, "Comparing data mining methods on the vaers database," *Pharmacoepidemiology and Drug Safety*, vol. 14, pp. 601–609, 2005.

[20] M. T. Niu, D. E. Erwin, and M. M. Braun, "Data mining in the us vaccine adverse event reporting system (vaers): early detection of intussusception and other events after rotavirus vaccination," *Vaccine*, vol. 19, pp. 4627–4634, 2001.