Coincidence Detection: A Fast Method for Discovering Higher-Order Correlations in Multidimensional Data

Evan W. Steeg, Derek Robinson, Ed Willis
Molecular Mining Corporation,
c/o PARTEQ Innovations, Queen's University,
Kingston, Ontario K7L 3N6, Canada
steeg@qucis.queensu.ca

Abstract

We present a novel, fast method for association-mining in high-dimensional datasets. Our Coincidence Detection method, which combines random sampling and Chernoff-Hoeffding bounds with a novel coding/binning scheme, avoids the exhaustive search, prior limits on the order k of discovered associations, and exponentially large parameter space of other methods. Tight theoretical bounds on the complexity of randomized algorithms are impossible without strong input distribution assumptions. However, we observe sub-linear time, space and data complexity in tests on constructed artificial datasets and in real application to important problems in bioinformatics and drug discovery. After placing the method in historical and mathematical context, we describe the method, and present theoretical and empirical results on its complexity and error.

Getting information from a table is like extracting sunlight from a cucumber.

(H. Farquhar, "Economic and Industrial Delusions", 1891)

Introduction

The measured attributes of individual objects or events (in general, observations or data records) are said to be associated or correlated when they occur together in, or are simultaneously absent from, individual records more often than could reasonably be expected "by chance" if they were independent of one another. Statistical association is the signature of a systematic and structural constraint operating on the population or system being investigated. The states of the variables are coupled in a non-random way, and we therefore infer there must be a reason, some causal and constitutive agency or law that is acting to limit the possible combined states of the variables (the exponentially vast product space formed by multiplying the number of possible states of every variable by those of every other variable) to just that subspace corresponding to the combined states observed. The states might be the recorded states of the different components of a computer network or automated factory, the combinations of which are crucial in diagnosing a system crash. Or the states might be the different possible values for key demographic variables in a huge database of households, some combinations of which may predict important buying or voting patterns.

Thus a fundamental goal of statistical analysis has been to estimate the shape of the joint distribution, the location of the the bulk of the "probability mass". Where are the peaks, the "modes"? These will be the regularities, the "suspicious coincidences" that provoke the question "Why?", and may lead to the formulation of physical and deductive (as opposed to merely statistical) hypotheses. But before speculating about underlying mechanisms, we first must know where to direct our curiosity: what (or where) are the "coincidences" that require explanation? In the past this question was given little attention, presumably because we have never lacked for conspicuous regularities that need explaining. More to the point, before the computer age, data collections of such size with so many variables did not exist. Thus just noticing the regularities in the data rarely exceeded our perceptual grasp.

Having motivated the problem, we now state it formally. Assume that we are given a database of M *objects* $\bar{s}_i$, each of which is characterized by particular *values* $a_{ij} \in \mathcal{A}_j$ for each of N discrete-valued *variables* c_j ("c" for column). A particular value for a particular variable is an *attribute* and denoted $a_l@c_j$. We further assume that there is some "true" underlying probability distribution $p()$ which, for all orders $k = 1, 2, \ldots, N$ specifies the probabilities for each possible k-tuple of attributes. For example, for $k = 1$, we have $p(c_j) : \mathcal{A}_j \longrightarrow [0, 1]$.

Table 1 contains a sample dataset for $M = 6$ rows (objects, transactions, records) and $N = 6$ columns (variables, items, fields). Any multidimensional relational database can be represented in this simple two-dimensional format by using some finite alphabet $\mathcal{A}_j$ of discrete symbols to represent the different possible

col1	col2	col3	col4	col5	col6
A	B	C	D	E	F
W	U	C	V	E	G
Z	L	C	M	W	M
V	U	C	V	A	A
G	B	C	D	Z	Z
W	L	C	M	E	Z
	⇑		⇑		

Table 1: A toy dataset of $M = 6$ rows and $N = 6$ columns.

(discrete or discretized) values for each of the the N variables v_j, represented as column c_j. We use the term "attribute" to mean a particular value for a particular variable. (This usage differs from that of some groups, who use "attribute" for the variable itself and write of "attribute-value pairs". Other groups use the term "item" for our "attribute".) In this simplistic example, the alphabet is the same for each variable. It appears that columns 2 and 4 are correlated, with $(B@2, D@4)$, $(L@2, M@4)$ and $(U@2, V@4)$ as the associated attribute pairs.

There are essentially three possible goals of probability density estimation:

1. Estimation of the fully-specified, fully higher-order joint probability distribution: Estimate a probability density $q()$ that closely approximates $p()$ and specifies $q(\alpha) = q(a_1^\alpha @ c_1^\alpha, a_2^\alpha @ c_2^\alpha, \ldots, a_k^\alpha @ c_k^\alpha)$ for *all* k-tuples α of attributes.
2. Hypothesis testing, for particular hypotheses concerning *particular* variables or *particular* combinations of variables or values: For example, are the data consistent with the hypothesis that columns $c_1^\alpha, c_2^\alpha, \ldots, c_k^\alpha$ are independent? In database analysis, one sometimes knows in advance which combinations of variables are of interest in a particular application.
3. Feature detection, or "association mining": Detect the most suspicious coincidences. For example, given a real number $\theta \in [0, 1]$, return a list of all k-ary joint attribute patterns

$$\alpha = (a_1^\alpha @ c_1^\alpha, a_2^\alpha @ c_2^\alpha, \ldots, a_k^\alpha @ c_k^\alpha) \quad (1)$$

such that Prob(Observed(α) | Independent $(c_1^\alpha, c_2^\alpha, \ldots, c_k^\alpha), \mathcal{M}) < \theta$, for some Observed number of occurrences of the pattern α and some model $\mathcal{M}$ which underlies one's sampling and hypothesis testing method. That is, find all combinations of attributes which, under some sampling and counting mechanism, are observed to occur significantly more often than one would expect given only the marginal probabilities of the individual attributes.

It has been noted (Lewis 1959; Ku & Kullback 1969; Miller 1993) that one can approximate the full joint distribution by use of maximum entropy assumptions (mainly the inter-column independence assumption) plus inclusion of a relatively few higher-order joint probability terms which deviate from these assumptions. Hence, full joint modeling reduces to association mining.

One cannot perform probability density estimation or association mining by examination of all combinations of variables. To test all possible k-tuples of variables for "suspicious coincidences" requires at least $O(\binom{N}{k} \cdot M)$ computational steps. To do this for *all* $2 \leq k \leq N$ is an $O(M2^N)$ computation, because one has to enumerate the powerset of a set of N columns. This powerset expansion of all joint probability terms, known variously as the full Gibbs model (Miller 1993) or the Bahadur-Lazarsfeld expansion, makes any direct, exhaustive approaches infeasible for all but the most trivial datasets.

Over the last twenty years, several distinct alternative approaches to the detection of higher-order correlations in multidimensional datasets have emerged. One popular approach is to restrict in advance the width of correlations sought, typically to $k = 2$ for pairwise correlations or to $k = 1$ for no correlations at all. This was the standard until a few years ago in computational molecular biology (Staden 1984; Korber *et al.* 1993), for example.

If one knows or assumes the variables to be sequentially related, as in the speech recognition (Sankoff *et al.* 1983) and macromolecular sequence analysis applications, then one can consider correlations only among sequential neighbors. This is the idea behind both "n-gram" approaches and most grammar-based approaches. Several groups have reported significant success in modeling protein sequence families and continuous speech with Hidden Markov Models (HMMs) (Krogh *et al.* 1994). For some of the same reasons why HMMs are very good at aligning the sequences in the first place, using local sequential correlations, these methods are less useful for finding the important sequence-distant correlations in data that has already been partially or completely aligned. The phenomenon responsible for this dilemma, termed "diffusion", is examined in some detail by Bengio and Frasconi (1995). Essentially, a first-order HMM, by definition, assumes independence among sequence columns, given a hidden state sequence. Multiple alternative state sequences can in principle be used to capture longer-range interactions, but the number of these grows exponentially with the number of k-tuples of correlated columns.

Many neural network architectures and learning algorithms are able to capture higher-order relationships among their inputs (Becker & Plumbley 1996). MacKay's "density networks", for example, use Bayesian learning to build componential latent variable models (MacKay 1994). However, the combinatorial explosion of priors and hyper-priors that need to be set

may severely limit the application of this method.

Coincidence Detection: A Novel Association-Mining Method

Like HMMs and Gibbs models (Miller 1993), the MacKay approach would benefit from a fast preprocessing stage that could find candidate subsets of correlated observable variables and allow one to pre-set some of the priors (or HMM state transitions, or Gibbs potentials) accordingly. Our Coincidence Detection method is designed to get around the central obstacle to association mining: we do not want to specify or limit, *a priori*, the number of possible k-tuples of correlated columns, the width k of any of them, or the degrees of correlation involved; and yet we do not want to explicitly represent and process latent variables or parameters for the exponentially-many possible k-tuples. Therefore, the method must be able to recognize the occurrence of patterns that provide evidence for k-ary correlations *whenever* they arise, and to analyze such patterns *only* when they arise — rather than set up data structures for higher-order patterns that may not ever appear.

A *coincidence* is defined as the absolute association of two or more attributes across a subset of observations (records), whereby whenever one attribute is present in a data record, so too will be the other attributes in this coincidence set (cset), and whenever any of these attributes is absent from a record, so also will be the other attributes in the cset. A cset is therefore a k-tuple of attributes found to exhibit absolute association within a certain set of records. If the records are taken to be rows of a data matrix, and attributes appear in columns, and the occurrence ("incidence") of a particular attribute in a data record is marked by a 1 and its absence by 0, a cset would be any subset of columns all of whose binary incidence vectors are identical.

All else being equal, the smaller the subset of records in question, the more such coincidences we would expect to occur. We might also expect that csets will tend to be of greater arity or "width" (larger k-tuples) in smaller sets of records than would be the case in samples involving a greater number of observations, since with each additional record there are more opportunities for at least one attribute to fail to coincide with the other $k-1$ attributes making up a given cset. And, as one would expect, the smaller the sample size, the more likely it is that many of the "coincidences" will turn out to be spurious or accidental.

Absolute association, or coincidence, has two virtues that recommend it as a basis for statistical computations: (1) It is applicable to greater than pairwise correlations or interactions; and (2) It can be detected automatically – it is not necessary to test every possible k-tuple of attributes in order to determine which k-tuples are coincident. However, to be useful, a measure of statistical association should recognize degrees of association. Moreover, there needs to be a means of distinguishing spurious from significant coincidences. We can make good these omissions by resampling – taking numerous random subsets from the database of observations (sampling with replacement), and counting how many times, out of the total number of trials, particular csets occur. Over the course of the resampling trials many csets will be collected, and some will be found to recur in a large proportion of the samples. The candidate csets can be tested for statistical significance by comparing their expected frequencies (predicted from the known frequencies of their individual component attributes in the database) with the frequencies actually observed. Significance and confidence levels can be estimated by standard methods, e.g., Bernoulli's theorem, Chernoff-Hoeffding bounds, or mutual information (Kullback-Liebler distance).

The coincidence detection procedure rests on the simple principle that global and partial association can be inferred from local and exact matches in random subsets of the data set. Inferring the probability of an attribute in a larger population from its observed frequency in many randomly chosen smaller subpopulations is of course the foundation of classical sampling theory. Coincidence detection differs insofar as its interest is with finding "knots" of interacting attributes that might only appear in a relatively small proportion of the data records but whose mutual correlation is, with respect to their expected correlation, surprisingly high.

Outline of Procedure

One simple variant of the Coincidence Detection method has four basic components:

Representation: The occurrences of an attribute in a set of records are summarized in a binary *incidence vector*. An incidence vector of length r has a 1 in the ith position iff the corresponding attribute, e.g., $B@2$, occurs in the ith record in the set.

Sampling: Take r records at a time, from a uniform distribution.

Binning, and Coincidence Detection: For each sampling iteration, throw the attributes into bins, according to their incidence vectors. These vectors act like r-bit addresses into a very sparse subset of 2^r address space. All the attributes in one bin constitute a *coincidence set*, or *cset*. Record the cset and the number $h : 0 \leq h \leq r$ of occurrences. (Note that h is the number of 1's in the incidence vector "address".)

Hypothesis Tests: After T iterations of sampling and binning, compare the observed number of occurrences of each cset with the number expected under the null hypothesis of statistically independent columns. The basis for the "expected" part of the hypothesis test is the probability of a match, or coincidence, of size h in a given r-sample for a cset α, as defined in Equation (1):

$$f_{match}(\alpha, h, r)$$

$$= \frac{r!}{h!(r-h)!} p(\alpha)^h p(\tilde{a}_1^\alpha @ c_1^\alpha, \ldots, \tilde{a}_k^\alpha @ c_k^\alpha)^{r-h},$$

where the joint probability terms reflect the independence assumption:

$$p(a_1^\alpha @ c_1^\alpha, \ldots, a_k^\alpha @ c_k^\alpha) = \prod_{l=1}^{k} p(a_l^\alpha @ c_l^\alpha)$$

$$p(\tilde{a}_1^\alpha @ c_1^\alpha, \ldots, \tilde{a}_k^\alpha @ c_k^\alpha) = \prod_{l=1}^{k} (1 - p(a_l^\alpha @ c_l^\alpha)),$$

and $\tilde{a}$ means the appearance of any symbol other than a. A Chernoff-Hoeffding bound (Hoeffding 1963) is used to implement the hypothesis testing, and so our procedure produces an estimate of the probability p^* of seeing n_{obs} occurrences of a cset α when the marginal probabilities of the components, and the independence assumption, predict only n_{exp} occurrences. Finally, the list of observed csets is sorted by their p^* values, and the procedure returns a small list of only the most "interesting" or "surprising" higher-order features, e.g., those which have $p^* < 0.001$.

How does one go about deciding whether or not to accept a cset as suspicious, on the basis of T, n_{obs} and n_{exp}? Let random variable X_i hold the value h_i for each iteration i, and let $X = \sum_{i=1}^{T} X_i$, and note that $0 \leq X \leq T \cdot r$. The method of Chernoff-Hoeffding bounds (Hoeffding 1963) provides the following theorem:

Let $X = X_1 + X_2 + \cdots + X_n$ be the sum of n independent random variables, where $l_i \leq X_i \leq u_i$ for reals l_i ("lower") and u_i ("upper").

Then

$$Prob[X - \mathbf{E}[X] > \delta] \leq \exp\left(\frac{-2\delta^2}{\sum_i (u_i - l_i)^2}\right). \quad (2)$$

For our purposes, we set $n = T$ and $l_i = 0$ and $u_i = r$ for all $i = 1, 2, \ldots, T$, and we thereby obtain

$$Prob[n_{obs} - n_{exp} > \delta] \leq \exp\left(\frac{-2\delta^2}{Tr^2}\right). \quad (3)$$

A pictorial representation of the main steps in the algorithm, using the example introduced in Table 1, is presented in Figure 1: two iterations of the r-sampling (for $r = 3$) on the toy dataset are depicted, top to bottom. For each iteration, the left-hand box represents the dataset, with outlined entries representing the sampled rows. The right-hand-box represents the set of bins into which the attributes collide. For example, in the first iteration, $B@2$ and $D@4$ both occur in the first and second of the three sampled rows, so they each have incidence vector 110 and collide in the bin labelled by that binary address. Bins containing only a single attribute are ignored; and "empty" bins are never created at all. All bins are cleared and removed after each iteration, but collisions (coincidences) are recorded in the *Csets* global data structure. Arrows indicate coincidences that involve the known-correlated columns 2 and 4.

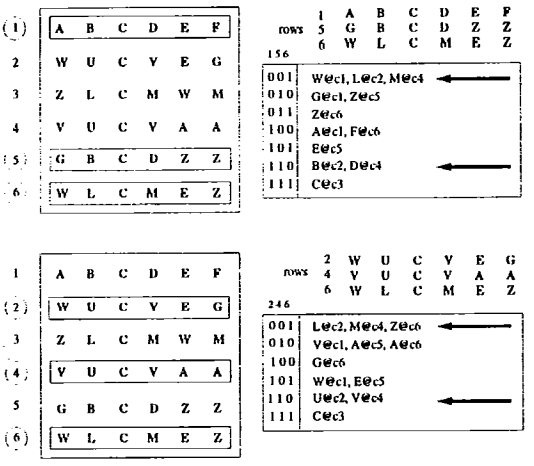

Figure 1: Operation of the Coincidence Detection Method.

An outline of the basic coincidence detection procedure is shown in Figure 2.

Complexity and Error Estimates

The coincidence detection procedure described above is a heuristic, randomized probabilistic approximation algorithm. Like all such algorithms, it does not lend itself to easy analysis, because many of the operations depend on the particular distribution of objects in the dataset. In this section, we estimate some bounds on the time, space, and data complexity of the method in terms of two distribution-dependent parameters L and T. We then present derivations and arguments for particular estimates of the expected values of those crucial parameters under some very broad distributional assumptions. Ultimately, however, precise analyses of the behavior of the method can be formulated only from empirical testing on a wide range of real and synthetic datasets from the various application domains. Some such testing is reported below.

Sample Complexity

The Chernoff-Hoeffding formulation of our hypothesis tests provides simple insights into the *sample complexity* of the procedure. Suppose some desired confidence level P^* is fixed in advance; by algebraic rearrangement of Inequality 3 given above, and by using the fact that $\mathbf{E}[X] = T \cdot \mathbf{E}[h]$, we obtain estimates for the necessary number T^* of r-samples:

$$T^* = \frac{-\log P^* r^2 + 4X\mathbf{E}[h]}{4\mathbf{E}[h]^2} + \frac{(-\log P^* r^2(-\log P^* r^2 + 8X\mathbf{E}[h]))^{1/2}}{4\mathbf{E}[h]^2} \quad (4)$$

```
Procedure to find suspicious coincidences:
0. begin
1.   read(DATASET);
2.   read(R, T);
3.   compute_first_order_marginals(DATASET);
4.   csets := {};
5.   for iter = 1 to T do
6.     sampled_data := rsample(R, DATASET);
7.     attributes := get_attributes(sampled_data);
8.     all_coincidences := find_all_coincidences(attributes);
9.     for coincidence in all_coincidences do
10.      if cset_already_exists(coincidence, csets)
11.        then update_cset(coincidence, csets);
12.        else add_new_cset(coincidence, csets);
13.      endif
14.    endfor
       /* Line 15 Optional */
15.*   csets := cull_uninteresting_csets(csets);
16.  endfor
17.  for cset in csets do
18.    expected :=
             compute_expected_match_frequency(cset);
19.    observed := get_observed_match_frequency(cset);
20.    stats :=
        update_stats(cset,hypoth_test(expected,observed));
21.  endfor
22.  print_final_stats(csets, stats);
23. end
```

Figure 2: The logical structure of a simple variant of the Coincidence Detection method is illustrated above.

T^* therefore grows with the negative log of the confidence level, which in our formulation is given in terms of the likelihood of the observed data given the inter-column independence assumption. This logarithmic sample complexity is fairly typical for algorithms employing random sampling and tail probability bounds.

Time and Space Complexity

Straightforward analysis of the steps of the algorithm in Figure 2 suggests a reasonable estimate of the overall asymptotic time complexity of the method is $O(MN + T(rN + \log L_*) + L_T)$, where L_* is the average size of the csets table over the T iterations and L_T is the final size. The space complexity can be estimated at $O(MN + L)$, where L is shorthand for both L_* and L_T.

The numbers M and N are typically fixed for the particular application. The more interesting components in the complexity estimates are L and T. L depends crucially on the distribution and degrees of correlation of variables in the dataset. T depends on the user-defined desired levels of accuracy, as well as on M and N We explore the possible ranges of L and T in the following section.

Estimating E[L], the Expected Size of the Cset Table

Although trivial proofs of meaningless bounds are possible, it is not possible to prove tight bounds without data distribution assumptions. In some applications, such restrictive assumptions may be possible and justified. However, generally we must focus on empirical testing. (Please see (Steeg 1996) and forthcoming papers for deeper analysis.)

We performed a set of experiments to observe how the number of stored csets grows as a function of the number of iterations T and the number of variables N. An attempt was made to observe in particular the worst-case behavior, with the assumption that L is maximal when the columns in the dataset are all mutually independent. (Dependencies imply redundancy, meaning that the same coincidences occur often; whereas for independent columns the attribute coincidences occur haphazardly and we would therefore expect more distinct coincidences to occur and hence more csets to be stored over the many iterations of sampling and binning. Alphabet size also plays a role: smaller alphabets tend to produce fewer, wider csets).

Four datasets of independent columns were generated and tested. The datasets had 50, 100, 200, and 500 columns. Each dataset contained 1000 rows and each column in every dataset was generated independently from a reasonable non-uniform distribution. Tests were also performed on a database of HIV protein sequences, as part of a protein structure prediction project described in (Steeg 1996; Steeg & Pham 1998).

Figure 3 shows a plot of the growth of L as a function of iteration number t, for $N = 200$, $r = 7$ and $T = 10,000$ iterations. A linear function, interpolated from a $t \approx 100$ neighborhood within the run, is also plotted for reference and comparison with the plot of $L(t)$.

The important result observed in Figure 3 is the sublinear growth of L. One might well imagine worst-case scenarios involving an exponential explosion of csets. Not only is the growth in csets better than exponential, it is better than linear. This would seem to bode well for practical application of coincidence detection procedures. As expected, the cset growth functions for these pathological datasets are closer to linearity than we observe for the real-world datasets that we have examined. The reader may check this by examining Figure 4, which pertains to the HIV datasets. Further discussion of this is found in (Steeg 1996).

Another important result is the sublinear growth of L as a function of N, the number of columns. Figure 5 illustrates clearly that exponential blowup and even "linear blowup" scenarios are overly pessimistic.

Analysis of Error in the Method — Types, Probability, and Bounds

There is a complex space of tradeoffs linking the size r of samples, number T of samples, and the relative

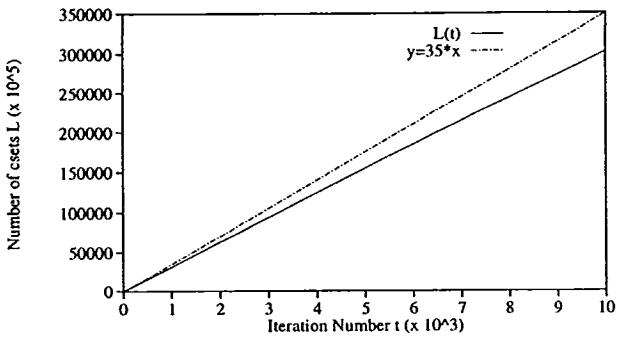

Figure 3: This plot illustrates $L(t)$, the size of the Csets data structure in our coincidence detection procedure, as a function of the number of r-sample iterations, for a dataset consisting of 200 independent columns. The parameter setting $r = 7$ was used. A linear plot is shown for comparison.

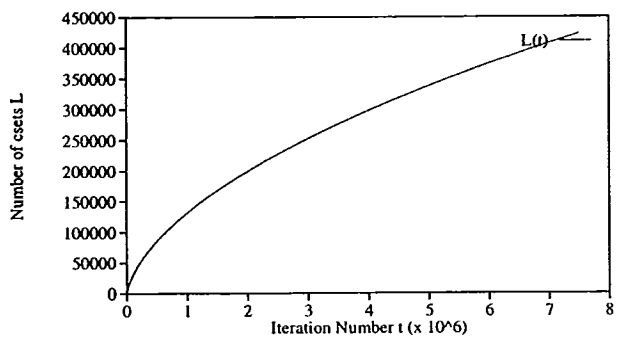

Figure 4: Plotted above is $L(t)$ for the HIV protein sequence dataset. $r = 7$ was used for this experiment.

risks of different types of error. First, intuitively: the larger the number T of samples, the smaller is the risk of overlooking the occurrences of a particular joint symbol occurrence. The smaller the sample size r, the smaller is this risk, too. Of course, a high level or small granularity of sampling has a cost: large T and small r raise the expected time and space complexity because they increase the expected size of the stored table of provisionally accepted csets.

In order to explore such issues more rigorously, it is first necessary to understand the coincidence detection procedure in terms of the three distinct levels of random sampling:

Level 1: The database itself: We may consider the M records to have been drawn from some larger underlying population of M_{all} objects. Alternative assumptions include larger but finite M_{all}, infinite M_{all}, and the case wherein $M_{all} = M$ (i.e., the database is all there is).

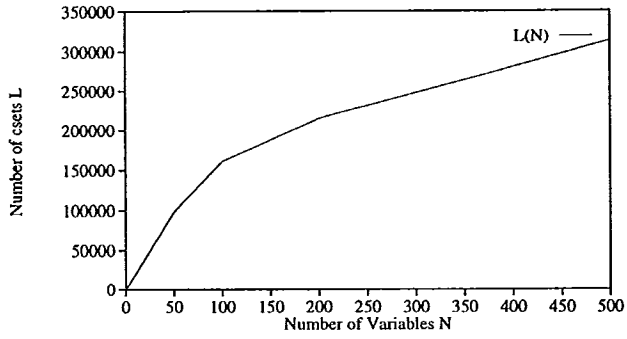

Figure 5: Plotted above is $L(N)$ for N Independent Columns. Sub-linear growth of L, the number of stored csets, as a function of N, the number of column variables, is observed when the number of iterations $T = 10,000$ is held constant.

Level 2: The r-sampling: r objects are drawn, randomly and identically distributed, without replacement, from the M objects in the database.

Level 3: The T iterations: The outer loop of the procedure iterates over many r-sample events, and records information on the outcomes. In the currently-described version of the algorithm, these r-sample draws are made, independently and identically distributed, with replacement. The process is modeled by a multinomial distribution.

Error may be injected at any or all levels of the procedure, and this greatly complicates any error analysis. In particular, the method requires and makes estimates of probabilities at the first and third levels of sampling: estimates of first-order marginal probabilities of attributes are made at Level 1; and estimates of the total count of joint attribute coincidences are made at Level 3. In practice it is often convenient to make a few explicit simplifying assumptions, one of which is the treatment of the error at Level 1 as independent from error at Level 3. For example, in assessing the error in higher-order joint probability estimates due to the iteration of r-sampling and binning, one might assume that the single-attribute marginal probabilities input to the procedure are correct. Our sampling procedures are as vulnerable to the problems of truly small dataset sizes as any other estimation procedure; we are currently developing a formulation which treats the Coincidence Detection output as a simple statistic within a Bayesian framework that provides for principled, optimal small-sample corrections and automatic selection of values for parameters r and T.

False Negatives: The *Masking* Problem The version of the coincidence detection method described thus far is particularly prone to a specific kind of false nega-

tive error. While the expected number of coincidences for a given cset α predicts the number of *all* occurrences of the α, the procedure's count of *observed* coincidences reflects only those instances where the component attributes of α occur *alone* together. That is, if α is defined as in Equation(1), then an observed bin containing just α gets counted towards α's occurrence total, whereas a bin containing α along with $a'@c'$ does not. Therefore, the support for α's "suspiciousness" is underestimated. For a rough estimate of the severity of the problem, consider the following simplified scenario. A dataset with a cset α and all other $n = N-k$ columns independent, is r-sampled for T iterations. Suppose α occurs as a coincidence in a particular iteration. Hence it has a binary incidence vector of length r with some number h of 1's and $r-h$ 0's. Let us pretend for the sake of simplicity that we have only binary attributes and let us ignore different individual attribute probabilities — this will not significantly affect the asymptotic results derived. The probability that some other attribute a will have an incidence vector exactly matching that of α is $p_{match} = \frac{1}{2^r}$. The probability that it will not match is $p_{no} = 1 - \frac{1}{2^r}$, and the probability that no such attribute a will co-occur exactly with α in this r-sample iteration is therefore $p_{none} = (1 - \frac{1}{2^r})^N$. Finally, the probability that *some* other attribute or attributes will coincide with α in this iteration is

$$p_{some} = 1 - \left[(1 - \frac{1}{2^r})^n\right]. \quad (5)$$

This estimate is sobering; it means that the proportion of coincidences that we are blocked from observing, with the simple version of coincidence detection presented in this thesis, grows very fast as the number of columns grows. These are illustrated in Figure 6. Note that the expansion of $(1 - \frac{1}{2^r})^n$, followed by a few simple algebraic steps, makes it clear that the p_{some} is dominated by a term $\frac{n}{2^r}$. This tells us, for example, that if we set $r = 10$, then $p_{some} \approx \frac{n}{1000}$, and so if $N > 500$ variables, then a particular interesting cset may be missed in about one half of all samples. One would do well to use a bigger r value and/or perform more sampling. Another solution is to maintain the Csets table in a lattice structure, so that increments made to the stored count for a cset are also propagated to its lower-dimensional component csets. Again, the theoretical worst case of exponential blow-up makes its appearance, though this scheme may be efficient in practice.

The good news is that the rate of growth of this masking problem is dependent upon the sample-size parameter r: higher r makes for less masking, and hence fewer extra samples are required to separate signal from noise — though one has to ensure the $r << M$. This relationship of result quality to choice of r is apparent in our empirical testing thus far. In our tests on the specially-constructed data, and on the HIV protein data, the masking problem has not obstructed us unreasonably for the values of r and N used.

Discussion and Future Work

Modelers of very large data sets, (e.g., census data or text corpora) are thwarted in their attempts to compute very far into a fully higher-order probabilistic model by both the computational complexity of the task and by the lack of data needed to support statistically significant estimates of most of the higher-order terms. One reasonable and common solution to this problem is to compute only a *subset* of higher-order probabilities, and extract a limited selection of higher-order features for construction of a database model. That is our approach. We suggest that efficient use can be made of limited computing resources by pre-selecting sets of higher-order associations using the coincidence detection algorithm described in this paper, and building the most significant into query tools and model-based classifiers and predictors based on existing statistical, rule-based, neural network or other methods.

A unique strength of the Coincidence Detection method is that it can discover, e.g., 41-ary correlations in the same time it takes to find pairwise correlations of equal statistical significance. This is in contrast with another class of association mining algorithms (Agrawal *et al.* 1996; Toivonen 1996) that has provided rule-based system developers with a way to attack large multidimensional datasets. These other methods build wide associations (large k) incrementally from narrower ones. Our method provides an interesting alternative for more exploratory analysis and discovery. An analogy may be helpful here: One set of methods (Agrawal *et al.* 1996; Toivonen 1996) is like those image compression and transmission methods which make a downloaded image appear on your screen one line at a time, top to bottom. Our method, on the other hand, is more akin to image transmission wherein the whole picture appears on your screen immediately, but at a low resolution which becomes clearer as the downloading proceeds. Another difference is that these incremental methods are designed to discover the *most frequent* associations, in which, for example, $p(A), p(B)$, and $p(A, B)$ are all high; whereas our Coincidence Detection method is designed to discover the *most surprising* associations, in which $p(A, B) \gg p(A)p(B)$. It must be noted that additional analysis is required to turn k-ary associations into directional rules of the form $A \rightarrow B$, and to turn an initial set of rules into an optimal ruleset. However, such analysis is made much more tractable once the essential higher-order structure of the data – the cliques of correlated variables – is discovered.

Our method also requires no assumptions about the number, size or sequential separation of the hidden higher-order features in the data. However, the relative advantage of our method is greatest on datasets in which there exist very strong and significant inter-attribute correlations, of whatever widths $2 \leq k \leq N$. The method is at a disadvantage when applied to data

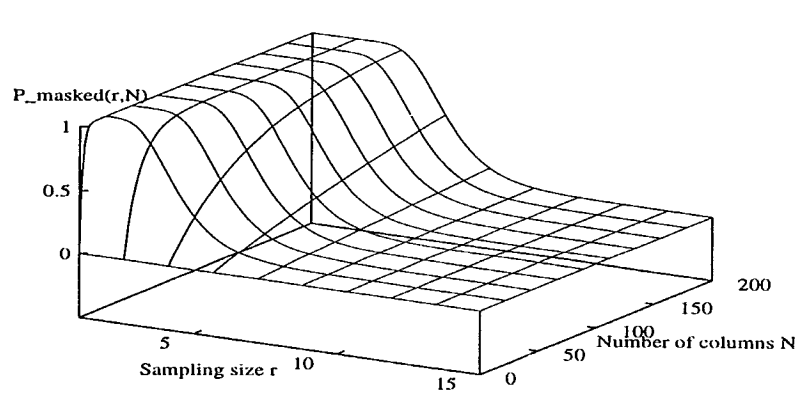

Figure 6: Shown above is a 3D plot of the estimated probability of a typical cset (correlated k-tuple of attributes) being masked in a given r-sample, as a function of r and N. Note that for higher values of r the probability of this particular kind of error is minimized.

with no correlations, or very weak ones; in such cases, the number of sampling iterations needed before correlations are detected, or before they can be ruled out, may be prohibitive.

The simplest, most naive version of the procedure was shown to perform well in practice at finding highly-correlated attributes, with observed time and space complexity better than linear in M and N. Further analysis, on a wide variety of different datasets with very different underlying probability distributions should produce a richer collection of results, providing a deeper understanding of the relative merits of various procedures and assumptions in various applications. We are currently applying our methods to protein sequence analysis and structure modeling, as well as to several other applications in medicine, business and engineering. We are also developing very fast parallel hardware implementations, in general and special-purpose configurations.

Acknowledgements

The authors wish to thank Hai Pham for programming and testing some versions of the Coinicidence method, and Laurie Ricker for timely editing and typesetting assistance.

References

Agrawal, R.; Mannila, H.; Srikant, R.; Toivonen, H.; and Verkamo, A. I. 1996. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining.* American Association for Artificial Intelligence. 307–328.

Agrawal, R.; Piatetsky-Shapiro, G.; and Padhraic, S. 1996. From data mining to knowledge discovery: An overview. In *Advances in Knowledge Discovery and Data Mining.* American Association for Artificial Intelligence. 1–34.

Becker, S., and Plumbley, M. 1996. Unsupervised neural network learning procedures for feature extraction and classification. *International Journal of Applied Intelligence* 6(3).

Bengio, Y., and Frasconi, P. 1995. Diffusion of context and credit information in markovian models. *JAIR* 3:249–270.

Bourlard, H., and Wellekens, C. J. 1986. Connected speech recognition by phonemic semi-markov chains for state occupancy modeling. In Young, I. T., ed., *Signal Processing III: Theories and Applications.* Elsevier Science Publishers B.V.

Fayyad, U. M.; Piatetsky-Shapiro, G.; Smyth, P.; and Uthurusamy, R., eds. 1996. *Advances in Knowledge Discovery and Data Mining.* American Association for Artificial Intelligence.

Fayyad, U. M.; Djorgovski, S. G.; and Weir, N. 1996. Automating the analysis and cataloging of sky surveys. In *Advances in Knowledge Discovery and Data Mining.* American Association for Artificial Intelligence. 471–493.

Hoeffding, W. 1963. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association* 58(301):13–30.

Korber, B.; Farber, R.; Wolpert, D.; and Lapedes, A. 1993. Covariation of mutations in the V3 loop of HIV-

1: An information-theoretic analysis. *Proc. Nat. Acad. Sci.* 90.

Krogh, A.; Brown, M.; Mian, I. S.; Sjolander, K.; and Haussler, D. 1994. Hidden Markov models in computational biology: Applications to protein modeling. *J. Mol. Biol.* 235:1501–1531.

Ku, H. H., and Kullback, S. 1969. Approximating discrete probability distributions. *IEEE Transactions of Information Theory* IT-15(4):444–447.

Lewis, P. M. 1959. Approximating probability distributions to reduce storage requirements. *Information and Control* 2:214–225.

MacKay, D. J. 1994. Bayesian neural networks and density networks. In *Proceedings of Workshop on Neutron Scattering Data Analysis.*

Miller, J. W. 1993. *Building Probabilistic Models from Databases.* Ph.D. Dissertation, California Institute of Technology, Pasadena, California.

Moore, A., and Lee, M. S. 1998. Cached sufficient statistics for efficient machine learning with large datasets. *JAIR* 8:67–91.

Sankoff, D.; Kruskal, J. B.; Mainville, S.; and Cedergren, R. J. 1983. Fast algorithms to determine RNA secondary structures containing multiple loops. In Sankoff, D., and Kruskal, J. B., eds., *Time Warps, String Edits, and Macromolecules: The Theory and Practise of Sequence Comparison.* Addison-Wesley.

Staden, R. 1984. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Research* 12:505–519.

Steeg, E. W., and Pham, H. 1998. Application of a novel and fast information-theoretic method to the discovery of higher-order correlations in protein databases. In *Proceedings of the Pacific Symposium on Biocomputing.*

Steeg, E. W. 1996. *Automated Motif Discovery in Protein Structure Prediction.* Ph.D. Dissertation, Department of Computer Science, University of Toronto.

Toivonen, H. 1996. Sampling large databases for association rules. In *Proceedings of the 22nd International Conference on Very Large Data Bases*, 134–145.