ABSTRACT

Motivation: Disease state prediction from biomarker profiling studies is an important problem because more accurate classification models will potentially lead to the discovery of better, more discriminative markers. Data mining methods are routinely applied to such analyses of biomedical datasets generated from high-throughput 'omic' technologies applied to clinical samples from tissues or bodily fluids. Past work has demonstrated that rule models can be successfully applied to this problem, since they can produce understandable models that facilitate review of discriminative biomarkers by biomedical scientists. While many rule-based methods produce rules that make predictions, they typically do not quantify the uncertainty in the validity of the rule itself. This article describes an approach that uses a Bayesian score to evaluate rule models.

Results: We have combined the expressiveness of rules with the mathematical rigor of Bayesian networks (BNs) to develop and evaluate a Bayesian rule learning (BRL) system. This system utilizes a novel variant of the K2 algorithm for building BNs from the training data to provide probabilistic scores for IF-antecedent-THEN-consequent rules using heuristic best-first search. We then apply rule-based inference to evaluate the learned models during 10-fold cross-validation; performed two times. The BRL system is evaluated on 24 published 'omic' datasets, and on average it performs on par or better than other readily available rule learning methods. Moreover, BRL produces models that contain on average 70% fewer variables, which means that the biomarker panels for disease prediction contain fewer markers for further verification and validation by bench scientists.

Contact: vanathi@pitt.edu

Supplementary information: Supplementary data are available at Bioinformatics online.

Received on April 9, 2009; revised on December 11, 2009; accepted on January 5, 2010

1 INTRODUCTION

High-throughput 'omic' data that measure biomarkers in bodily fluids or tissues are accumulating at a rapid pace, and such data have the potential for the discovery of biomarkers for early diagnosis, monitoring and treatment of diseases such as cancer. Data mining methods that learn models from high-dimensional data are being increasingly used for the multivariate analyses of such biomedical datasets. Together with statistical univariate analyses, some insights into predictive biomarkers of disease states can be gleaned, though the results may not generalize due to the small sizes of available training data, typically less than 200 samples.

Due to the large imbalance between variable dimensionality (several thousand) and the sample size (a few hundred), there is a need for data mining methods that can discover significant and robust biomarkers from high-dimensional data. Rule learning is a useful data mining technique for the discovery of biomarkers from high-dimensional biomedical data. We have previously developed and applied rule learning methods to analyze 'omic' data successfully (Gopalakrishnan et al., 2004, 2006; Ranganathan et al., 2005). Rules have several advantages, including that they are easy for humans to interpret, represent knowledge modularly and can be applied using tractable inference procedures.

In this article, we develop and evaluate a novel probabilistic method for learning rules called the Bayesian rule learning (BRL) algorithm. This algorithm learns a particular form of a Bayesian network (BN) from data that optimizes a Bayesian score, and then translates the BN into a set of probabilistic rules. The use of the Bayesian approach allows prior knowledge (as probabilities) to be incorporated into the learning process in a mathematically coherent fashion. The possibility of over-fitting is attenuated by the incorporation of prior probabilities into the rule-discovery process. BRL outputs the predictive rule model with the best Bayesian score, which represents the probability that the model is valid given the data.

The remainder of the article is organized as follows. Section 2 presents the BRL algorithm and briefly reviews other popular rule learning methods. Section 3 describes the datasets and the experimental setup to evaluate BRL. Section 4 presents the results of applying BRL to 24 published 'omic' datasets, and compares its performance with multiple rule-learning algorithms. Section 5 presents our conclusions.
Bayesian rule learning for biomedical data mining

Fig. 1. A BN structure learned by BRL on the acute lymphoblastic leukemia (ALL) versus acute myeloid leukemia (AML) dataset (Golub et al., 1999) and its equivalent set of rules. #Class refers to the target variable that can take on values of either 0 (ALL) or 2 (AML) in this example. Each rule is associated with statistics from the training data (see note 1).

score. This score is used for model selection. We now discuss in detail the BRL method. This algorithm learns BN models from the data and the model is then translated into a set of rules with associated statistics.1 These rules are mutually exclusive and exhaustive over the values of the predictor variables, and hence inference using these set of rules becomes trivial. Given a new test case, the rule that matches its values for the predictor variables is used to infer the value of the target variable.

2.1 Bayesian networks

A BN is a probabilistic model that consists of two components: a graphical structure and a set of probability parameters. The graphical structure consists of a directed acyclic graph, in which nodes represent variables and variables are related to each other by directed arcs that do not form any directed cycles. Associated with each node (let us call it a child node) is a probability distribution on that node given the state of its parent nodes, and all the probability distributions for all the nodes taken together provide a factored probability distribution for the joint value state denoted by index k. The values or expression of these genes influence the ability to improve upon greedy search with a beam size of 1.

Using the constrained structure of the algorithm in Figure 2, as illustrated by the model shown in Figure 1, Equation (1) simplifies further to the following equation:

\[ P(D|M) = \prod_{i=1}^{q} \frac{(r_i - 1)!}{N_i r_i - 1} \prod_{j=1}^{k} N_{ijk}. \]  

(2)

where q is the number of joint parent states of the target variable, r the cardinality (number of values or states) of the target variable and N_p the number of instances in the training database D for which the target variable has the value k and its parents have the joint value state denoted by index j. Also, N_ijk is the sum over k of N_ijk.

Figure 1 depicts an example of BN structure learned by BRL, where #Class represents a target variable and two genes M23197_at and U64699_at are its parents. The values or expression of these genes influence the target class. Figure 1 depicts M and the set of rules derived from M. A rule set is defined as the conditional probability, such as the conditional probability P(#Class | M23197_at, U64699_at) for all values of M23197_at.

and several methods have been developed to automatically learn BNs from data (Neapolitan, 2004). Here, we use the Bayesian method called K2 (both the K2 scoring measure and the K2 forward stepping search heuristic) for learning BNs (Cooper and Herskovits, 1992).

The K2 scoring measure (Cooper and Herskovits, 1992) assumes that the variables are discrete, the cases (training examples) occur independently, and there are a uniform prior probability distribution over the space of all possible network structures. The K2 measure also assumes that every possible probability distribution over the values of a node given a state of its parents is equally likely (uniform). Under these assumptions, a closed form solution for the Bayesian score is given by the following equation (Cooper and Herskovits, 1992):

\[ P(D|M) = \prod_{i=1}^{q} \frac{(r_i - 1)!}{N_i r_i - 1} \prod_{j=1}^{k} N_{ijk}. \]  

(1)

where M is the BN structure under consideration, D the data used to learn M, q the number of variables in M, r_i the number of parent states of child variable i, r_j the cardinality (number of values or states) of variable j, and N_p the number of instances in the training database D for which variable i has the value k and the parents of i have the value state denoted by index j. Also, N_ijk is the sum over k of N_ijk.

Since it is usually computationally intractable to search over all possible BN structures to locate the one with the highest score, a greedy search in the space of BNs is employed. The greedy search method used by K2 (Cooper and Herskovits, 1992), requires an ordering on the variables and a user-specified parameter for the maximum number of parents any variable in the network can have.

2.2 Bayesian rule learning

The BRL algorithm uses the score given by Equation (1) to learn simple BNs. In particular, (i) BRL considers constrained BNs where the BN consists of one child variable, which is the target to be predicted, and other variables are parents of it; (ii) only the target node is evaluated with the Bayesian score; and (iii) models are searched by utilizing a beam (memory of particular size) to store high-scoring BNs. The beam refers to a restricted memory size for storing BN structures and is implemented as a priority queue of fixed width (beam size), where BNs are stored according to their score. This reduces the memory requirement for heuristic, best-first search, by exploring only those BN structures that are high scoring, while at the same time providing the ability to improve upon greedy search with a beam size of 1.

\[ P(D|M) = \prod_{i=1}^{q} \frac{(r_i - 1)!}{N_i r_i - 1} \prod_{j=1}^{k} N_{ijk}. \]  

(2)

where q is the number of joint parent states of the target variable, r the cardinality (number of values or states) of the target variable and N_p the number of instances in the training database D for which the target variable has the value k and its parents have the joint value state denoted by index j. Also, N_ijk is the sum over k of N_ijk.

Figure 1 depicts an example of BN structure learned by BRL, where #Class represents a target variable and two genes M23197_at and U64699_at are its parents. The values or expression of these genes influence the target class. Figure 1 depicts M and the set of rules derived from M. A rule set is defined as the conditional probability, such as the conditional probability P(#Class | M23197_at, U64699_at) for all values of M23197_at.

1Missing values can be accommodated by including an extra state for a missing value of a variable. That extra state can be labeled, for example, as ‘missing’. Thus, a variable with two domain values (e.g. true and false) becomes a variable with three possible values (e.g. true, false and missing).

1CF refers to certainty factor or degree of belief in the rule (Shortliffe et al., 1975; Heckerman, D., 1985). P, F-value from Fisher’s exact test; TP, number of true positives (match both sides of the rule); FP, number of false positives (match rule antecedent, but not consequent); Pos, number of positive examples (match rule consequent); and Neg, number of negative examples (do not match rule consequent). Cost measures could be incorporated, but are not used for the experiments in this article.
INPUT: Discrete predictor variables $\{X, x\}$ and target variable $T$, an upper bound $\text{MAX}\_\text{CONS}$ on the number of parents that $T$ can have, beam-width $b$, and training data $D$ containing $m$ cases.

OUTPUT: A disjunction of conjunctive probabilistic IF-THEN rules.

DEFINITIONS:

- $M$: Bayesian Network structure.
- $\text{PD}(M)$: function that returns the Bayesian score (marginal likelihood) for model $M$ and data $D$.
- $B$: Beam of size $b$ that sorts models by their score in descending order.
- $V$: Set of all variables $X$.
- $F$: Priority queue containing final structures (that cannot be improved further by adding a single variable) sorted by their scores in descending order.
- $A$: Subset of $V$ containing $X$ already appearing in final structures.
- $\text{Parents}(M) = \text{function that returns the set of parents } X_i \text{ of } T \text{ in } M$.

ALGORITHM:

1. Create model $M$ containing just target node $T$ and place $M$ on beam $B$.
2. $A := \emptyset$.
3. WHILE (beam $B$ is not empty AND $A \subsetneq V$) DO:
   5. $X = Y - \{\text{parents}(M)\} \cup \{\text{NOT in } M \text{ or } A\}$.
   6. Set $\text{score} \_\text{improves} := \text{false}$.
       IF ($X$ tree empty AND $\{\text{parents}(M)\} = \text{MAX}\_\text{CONS} \text{ THEN}$)
       7. FOR each possible joint state $J$ DO:
           $M_{\text{new}} := \text{Add } X_i \text{ to parent } T \text{ in } M$
           IF ($\text{score}_M \_{\text{improves}} > \text{score}_M \text{ DO THEN}$)
               Place $M_{\text{new}}$ on $B$.
       ENDFOR:
   8. Set $\text{score}_\text{improves} := \text{true}$.
   9. $\text{ENDWHILE}$.
10. $M, m$: First model removed from priority queue $F$.
    FOR each possible joint state $J$ of values for all $X \not\in M$
    FOR each possible value $L$ of target variable $T$
    Calculate $\text{CF}(R, J)$.
    ENDFOR
    Let $s = \text{argmax}_k \text{score}_\text{improves}$.
    Output $R_s$: IF ($x$ in state $J$) THEN $\text{Rule } s$ with $\text{CF}(R_s)$.
    $\text{ENDFOR}$.

Fig. 2. The BRL algorithm.

and U46499_at. In the example, each of the two genes can take on two discrete range values of the predictors. Hence, the total number of possible combinations of the values for the predictor variables is four (Rules 0–3). The Bayesian score [Equation (2)] represents the joint probability of the model and the data under the assumption of uniform structure priors. Since $P(M|D) = P(D|M)$, given the assumption that all models are equally likely a priori, the Bayesian score can be directly utilized as a measure of model uncertainty and used to prioritize and select models. Breadth-first marker propagation (Aronis and Provost, 1997) is utilized to record the matching statistics (or counts) and greatly speeds up the calculations by requiring just one pass through the training data to record the counts. BRL is thus very efficient and runs in $O(n^2)$ time given $n + 1$ variables and $m$ training examples, using the default constant values for beam size $b$ and the maximum conjunct parameter $\text{MAX}\_\text{CONS}$, which are both user-specified.

The BRL algorithm is shown in Figure 2. It takes as input a set of input variables $X$, a target variable $T$, an upper bound on the number of parents that $T$ can have and training data containing vectors of values for $X$’s and the corresponding value for $T$. The user can also provide a beam width $b$ that restricts the number of $\text{BNs}$ that are in the priority queue. The default beam width is 1000.

In Step 1, a BN containing just the target node with zero parents is created, and it is scored using Equation (2). Step 2 initializes the list of variables $X$ that appear in models that have good scores and cannot have a better score by the addition of any single parent of $T$. The loop condition in Step 3 checks to see whether there are still models on the beam that can be expanded further by the addition of a parent variable such that the score would improve. Steps 4–8 perform one-step forward search by adding one more allowed variable as an additional parent of the target $T$ to the structure. Specialization refers to the addition of a parent variable (not already present) to the structure of current model, such that the total number of parent variables in the model does not exceed the upper bound $\text{MAX}\_\text{CONS}$. The default value we use for $\text{MAX}\_\text{CONS}$ is 5.

In Step 9, a check is made to see whether the score of the model removed from the beam improved after all one-step specializations. If not, that model is placed on a priority queue containing final model structures ordered according to their Bayesian scores. Even though we store many final models on the final priority queue, only the best scoring model is presented to the user in the form of a rule model in Step 10. For each value of target variable $T$, its probability given each state of possible values of its parent variables is calculated from the training data. The certainty factor is calculated as shown in Figure 3.

We perform a simple pruning in Step 10, whereas we output only the rule with the target value that has the highest probability given the particular state of its parent variables (Fig. 3). There are many other methods that could be used to perform pruning of the rules generated in Step 10, such as the number of training examples covered by the rule (Han and Kamber, 2006).

Also, we perform an optimization to increase search efficiency of the BRL algorithm. As can be seen in Step 9, the algorithm keeps track of those sets of variables that cannot be specialized further by addition of single variables as parents of the target such that the Bayesian score improves. The assumption is that if a predictor is in some final rule, then it is unlikely to be a strong predictor in another rule. While empirically, we observe that this assumption works well for most datasets that we have analyzed, it is certainly possible that there are datasets for which the assumption will not work well. Thus, extensions to this basic BRL algorithm can be explored along several directions to overcome some of the assumptions and limitations.

2.3 Rule learning methods

For our experiments we used three readily available rule learning methods: Conjunctive Rule Learner, RIPPER and C4.5. Conjunctive Rule Learner is a simple rule learner that learns a set of simple conjunctive rules that optimizes the coverage and predictive accuracy. It uses a technique called Reduced Error Pruning (Furnkranz and Widmer, 1994) to trim an initial set of rules to the smallest and simplest subset of rules that provide highest discrimination. Repeated Incremental Pruning to Produce Error Reduction (RIPPER) was developed by Cohen (1995) and uses the REP technique in Conjunctive Rule Learner, but performs multiple runs (Cohen, 1996). C4.5 is a decision tree learner developed by Quinlan (1994) that extends the basic decision tree learner ID3 (Quinlan, 1986) to improve classification. These improvements include parameterization of the depth of the decision tree,
rule post-pruning, ability to handle continuous-valued attributes, ability to choose the best attribute to use in growing the decision tree and an increase in computational efficiency (Gabrilovich and Markovitch, 2004; Xing et al., 2007).

2.4 Discretization
The rule learners described above require variables with discrete values. We used a new discretization method called heuristic efficient Bayesian discretization (EBD; Lustgarten, 2009), which we developed for transforming continuous data to discrete. EBD uses a Bayesian score to discover the appropriate discretization for a continuous-valued variable and runs efficiently on high-dimensional biomedical datasets. Compared with Fayyad and Irani’s (FI) discretization method (Fayyad and Irani, 1993), which is an efficient method commonly used for discretization, EBD had statistically significantly better performance in the evaluation report in Lustgarten et al. (2008).

3 EXPERIMENTAL SETUP
3.1 Biomedical datasets
The performance of BRL and the three comparison rule learning methods were evaluated on 24 biomedical datasets [21 publicly available genomic datasets, two publicly available proteomic datasets from the Surface Enhanced Laser/Desorption Ionization Time of Flight (SELDI–TOF; Wright et al., 1999) mass spectrometry platform and one University of Pittsburgh proteomic dataset, which is a diagnostic dataset from an Amyotrophic Lateral Sclerosis study obtained using the SELDI–TOF platform]. The datasets, along with their type (prognostic/diagnostic), number of instances, number of variables and the majority target-value proportions are given in Table 1.

3.2 Data mining techniques and statistical analysis
As mentioned, for comparison, we used three rule learners, namely, Conjunctive Rule Learner, RIPPER and C4.5 as implemented in WEKA version 3.5.6 (Witten and Frank, 2005). We used two versions of BRL, namely, BRL4 (beam size set to 1) and BRL1000 (beam size set to 1000). We used heuristic EBD for discretization; the discretization cutpoints were learned from the training set and then applied to both the training and test sets. We implemented EBD in Java, so that it can be used in conjunction with WEKA.

We evaluated the rule learning methods using 10-fold cross-validation performed two times. The methods were evaluated using two measures: balanced accuracy (BACC), which is the average of sensitivity and specificity over all one-versus-rest comparisons for every target variable, and relative classifier information (RCI; Sindhwani et al., 2001). These measures are described below.

The BACC differs from accuracy that it compensates for skewed distribution of classes in a dataset. BACC is defined as follows:

\[
\text{BACC} = \frac{\sum \text{Sensitivity}(c) \times \text{Specificity}(c)}{|C|}
\]

\[
\text{Sensitivity}(c) = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c}
\]

\[
\text{Specificity}(c) = \frac{\text{TN}_c}{\text{TN}_c + \text{FP}_c + \text{FN}_c}
\]

where \(C\) is the set of the target variable values, and \(\text{Sensitivity}(c)\) [\(\text{Specificity}(c)\)] refers to the sensitivity (specificity) of the target value \(c\) versus all other values of the target. \(\text{TP}_c\) is the number of samples predicted to have the value \(c\) for the target variable given that the observed value is \(c\), \(\text{FN}_{\neg c}\) is the number of samples predicted to have a value other than \(c\) for the target variable given that the observed value is not \(c\) and \(\text{FP}_{\neg c}\) is the number of samples predicted to have the value \(c\) for the target variable given that the observed value is not \(c\).

In the type (T) column, P signifies prognostic and D signifies diagnostic. #C represents the number of classes, #A the number of attributes within the dataset, #S the number of samples and M is the fraction of the data covered by the most frequent target value.

4 RESULTS
The average BACCs obtained from 10-fold cross-validation performed two times for each of the 24 datasets are shown in Table 2.
As can be seen from the average BACCs for the 24 datasets, both BRL1 and BRL1000 clearly perform better than the other rule learning methods. This holds for both the genomic datasets (1–21) and the proteomic datasets (22–24).

We see that BRL1000 has the highest BACC on 15 datasets, while BRL1 has the highest BACC on 4 datasets. On the remaining five datasets, C4.5 has the highest BACC on three, ties with BRL1 on one and Conjunctive Rule Learner has the highest on one. Only the first dataset is very easy to classify by all rule learners. As seen in Table 3, the performance of both BRL4 and BRL1000 are statistically significantly better than C4.5, its nearest competitor in terms of BACC. When compared with each other, BRL1000 outperforms BRL1.

The average RCIs obtained by the various rule learning methods are shown in Table 4. BRL1000 has the highest RCI on 19 datasets, whereas BRL1 has the highest RCI on 3 datasets. There was one tie among the two BRL methods and C4.5. In addition, C4.5 has the highest RCI on one dataset. In Table 5, we compare the difference in performance using the RCI measure between C4.5 with BRL1 and BRL1000 and both methods are statistically significantly better than C4.5; the difference in performance using the RCI measure between BRL1000 and BRL1 is statistically significant in favor of BRL1000. Table 6 depicts a comparison of the average number of variables (markers) appearing in the rule models for C4.5, BRL1 and BRL1000, when run with default parameter settings. The average was calculated over the models generated from 20 folds (obtained

672
Table 5. The statistical comparisons between C4.5 and the two BRL implementations using RCI

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Average</th>
<th>Diff.</th>
<th>r-test (t-score)</th>
<th>Wilcoxon (Z-score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRL1 versus C4.5</td>
<td>44.88</td>
<td>13.50</td>
<td>0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>C4.5</td>
<td>31.38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRL1000 versus C4.5</td>
<td>51.89</td>
<td>20.51</td>
<td>&lt;0.001</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>BRL1000</td>
<td>44.88</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C4.5</td>
<td>31.38</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6. Comparison of the average number of variables in the models produced by C4.5 and BRL over all folds

<table>
<thead>
<tr>
<th>#</th>
<th>C4.5</th>
<th>BRL1</th>
<th>BRL1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>3.00</td>
<td>3.00</td>
<td>3.00</td>
</tr>
<tr>
<td>3</td>
<td>5.00</td>
<td>4.00</td>
<td>5.00</td>
</tr>
<tr>
<td>4</td>
<td>7.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>5</td>
<td>6.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>6</td>
<td>6.00</td>
<td>2.00</td>
<td>2.00</td>
</tr>
<tr>
<td>7</td>
<td>5.00</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>8</td>
<td>7.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>9</td>
<td>5.00</td>
<td>3.00</td>
<td>4.00</td>
</tr>
<tr>
<td>10</td>
<td>5.00</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>11</td>
<td>12.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>12</td>
<td>4.00</td>
<td>3.00</td>
<td>3.00</td>
</tr>
<tr>
<td>13</td>
<td>8.00</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>14</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>15</td>
<td>7.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>16</td>
<td>30.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>17</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>18</td>
<td>26.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>19</td>
<td>18.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>20</td>
<td>15.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>21</td>
<td>88.50</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>22</td>
<td>32.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>23</td>
<td>27.00</td>
<td>5.00</td>
<td>5.00</td>
</tr>
<tr>
<td>24</td>
<td>7.00</td>
<td>3.00</td>
<td>3.00</td>
</tr>
<tr>
<td>Average</td>
<td>13.72</td>
<td>4.00</td>
<td>4.08</td>
</tr>
</tbody>
</table>

From stratified 10-fold cross-validation repeated two times) on the 24 datasets. As shown, the BRL models have ~70% less variables on average in their models than C4.5.

If each predictor variable has only two discretized ranges of values, then BRL with default parameters would generate between $2^2$ and $2^7$ rules on average. However, discretization could yield a larger number of value ranges for a variable, thereby increasing the number of rules generated by BRL. To reduce the number of rules, we can prune rules with zero coverage, that is, those rules whose left-hand side does not match any of the samples in the training data. We notice that pruning does not harm BRL’s performance. However, rule pruning could cause problems during testing, since rules that do not match training data could still match test data. We include an example of pruned rules and also C4.5 rules in the

Supplementary Material. The variables chosen in BRL’s predictive models are often different from those chosen by C4.5.

4.1 Discussion

There are several advantages that accrue from BRL that are not available in current rule learning algorithms. BRL allows for the evaluation of the entire rule set using a Bayesian score. Using such a score results in a whole model evaluation instead of a per rule (or local) evaluation, which often occurs with Ripper and C4.5. The Bayesian score allows us to capture the uncertainty about the validity of a rule set. BRL currently uses this score only for model selection. However, the score could be utilized in extensions to BRL for performing inference when rule sets can be weighted by this score, which would be a form of Bayesian model averaging.

A Bayesian approach allows incorporation of both structure and parameter priors. When training data are scarce, such as in ‘omic’ data analysis, it is useful to incorporate prior knowledge to improve the accuracy of learned models. For example, a scientist could define all of the variable relationships using either a knowledge base or restrict the possible variables with which to build the model (Frey et al., 2005; Miriam et al., 2005). In a Bayesian approach, a scientist might provide prior knowledge specifying conditional independencies among variables, constraining or even fully specifying the network structure of the BN. In addition to providing such structure priors, the scientist might also specify knowledge in the form of prior distributions over the parameter values of the model. Structure priors are arguably the most useful, however, because in our experience scientists are often more confident about structural relationships than about parameter values. We have not explored informative priors in this article. We used uniform parameter and uniform structure priors. Exploring informative structure priors in this domain is a direction for future research.

There are different ways of representing non-informativeness of parameters using the Dirichlet priors. We have explored one approach, it would be useful to explore other approaches as well. An interesting open problem is to investigate methods for BRL rule ordering and pruning within a set of rules. For example, pruning a set of BRL rules based on using local structure and scores (Chickering et al., 1997; Friedman and Goldszmidt, 1996; Visweswaran and Cooper, 2005) would be worth investigating.

A major advantage of BRL is that it can find models with fewer variables (markers) that have equivalent or greater classification performance than those obtained from several other rule learning methods. Fewer variables mean fewer markers for biological verification and subsequent validation. This is important in biomarker discovery and validation studies that have to be designed carefully and under tight resource constraints.

5 CONCLUSIONS

We have shown that using a BN approach to generate rule models not only allows a more parsimonious model (in terms of the number of variables), but also produces results that are statistically significantly superior to common rule learning methods. It also allows the creation of probabilistic rules that are optimized on the rule model level as opposed to the current method of evaluation per rule. Using BN as a
ACKNOWLEDGEMENTS

We thank the Bowser Lab at the University of Pittsburgh (Department of Pathology) for use of the proteomic dataset from Ranganathan et al. (2005). We thank Philip Ganchev, the University of Pittsburgh (Intelligent Systems Program) for help with additional verification runs of the experiments with BRL.

Funding: National Institute of General Medical Sciences (grant number GM071951); the National Library of Medicine (grant numbers T15-LM070759, R01-LM06696), the National Science Foundation (grant number IIS-0325581).

Conflict of Interest: none declared.

REFERENCES


