Learning Bayesian network by a Mesh of points

Byron Oviedo
Universidad Técnica Estatal de Quevedo
Faculty of engineering sciences
Quevedo, Los Ríos, Ecuador, EC120508
email: boviedo@uteq.edu.ec

Luis Moreira
Amilkar Puris
Pavel Novoa
Universidad Técnica Estatal de Quevedo
Faculty of engineering sciences
Quevedo, Los Ríos, Ecuador, EC120508
email: luismoreira@uteq.edu.ec
apuris@uteq.edu.ec
pnovoa@uteq.edu.ec

Serafin Moral
Universidad de Granada
Granada, España, 18071
email: smc@decsai.ugr.es

Abstract—In this work we applied Variable Mesh Optimization population metaheuristic (VMO) for Bayesian network (BN) structure learning as a score-and-search method. Our idea was to represent each node of the Mesh as a Bayesian network through a set of arcs. Then new BNs are created using (union and difference) operations among sets. For this process, three types of BNs are identified, local optima (BNs with the best score in each neighborhood), global optima (BN with the best score among local optima), and frontier solutions (most and least different in structure BNs). Finally the clearing process is applied to select the most representative BNs in the Mesh (score and structure). For determining the global score, each BN is used as a Bayesian classifier and classification accuracy is obtained using cross validation over dataset. The proposal is compared with other classifiers using UCI repository data set. Results show that our proposal obtains the best score, that proves to be a very competitive algorithm for supervised classification.

I. INTRODUCTION

Probabilistic graphical models and, in particular, Bayesian networks [?] are useful tools to represent and make inferences in problems with a high number of variables in presence of uncertainty. They provide models in which probabilistic knowledge can be efficiently represented, and though the problem of computing conditional probabilities for variables of interest is #P-complete there are efficient algorithms which are able of solving big size problems. For these reasons, they have successfully applied in may fields as medical diagnoses, troubleshooting analysis, gene expression analysis, or information retrieval [?].

A Bayesian network has two parts: a qualitative one consisting of a directed acyclic graph representing (in)dependence relationships between the problem variables and a quantitative one, consisting of a list of conditional probability distributions, which given the independences represented by the graph, determine a joint probability distribution for all the variables in the problem.

One of the reasons for the success of Bayesian networks is that they have a precise and clear semantics for the graphical part, in such a way that they can directly elicited from trained experts. But, perhaps it has been more important the development of algorithms to estimate or learn a Bayesian network from a dataset of observations [?]. There are even procedures able of learning from data and experts in an interactive way [?].

There are two main approaches for learning a Bayesian network from a set of observations: independence tests approaches [?] and score+search methods [?]. The first one is based on making statistical inference tests to detect the independence relationships in data that are represented by the graph. In the second one a score measuring the suitability of a graph given a dataset is proposed and the problem of finding the graph optimizing the score is solved. Some authors have also mixed ideas from both approaches as in [?], [?].

One problem associated to the use of score+search procedures is that the problem of finding the graph optimizing the score is NP-hard [?] with a search space (the set of all directed acyclic graphs (DAG) of a given size) which is super-exponential. As a consequence, many metaheuristic strategies have been proposed in order to find fast and near-optimum solutions [?], [?]. Among them, we can cite: Ant Colony Optimization (ACO) and [?], Particle Swarm Optimization. Evolutionary algorithms are the most usual strategy on this context. A review for these algorithms is presented in [?].

In previous works, we have presented Variable Mesh Optimization (VMO) [?], an efficient MH to explore search complex spaces. This algorithm uses a population mesh to explore the search space, through the application of several combination methods to create new solutions in the mesh (Expansion Process). Finally, it uses a clearing method [?] that maintains the most representative solutions (Contraction Process). VMO have been applied with competitive results on several optimization problems as Function Approximation [?], Niching Problems in Multimodal Function [?], Feature Selection problem [?] and Traveling Salesman Problem [?], showing that VMO is a competitive model.

The aim of this work is to expand the application space of VMO metaheuristic, applying it to structural learning of Bayesian networks. For this, some adaptations on the algorithm are studied and the results of our proposal are compared with others traditional Bayesian classifiers using UCI repository data sets.

This work is organised as follows: In Section ?? we describe
a background of the Bayesian networks and their use for supervised classification. Moreover, the main operations of the VMO metaheuristic are presented in this section. Then, in Section ??, some elements in VMO are adapted for the structure learning problem of Bayesian Network and its results are compared with others Bayesian classifiers. Finally, in Section ??, we present the main conclusions of this work.

II. BACKGROUND

A. Bayesian Networks

Assume that we have a set of variables $X = \{X_1, \ldots, X_n\}$, where each variable takes its values on a finite set $Val(X_i)$. A Bayesian network on these variables has two parts: a Directed Acyclic Graph (DAG) $G = (X, A)$, with a node for each variable $X_i \in X$ and where arcs represent (in)dependence relations between the variables according to the d-separation criterion [?]; and a list of conditional probabilities $P(X_i | \Pi(X_i))$ where $\Pi(X_i)$ is the set of parents of variable $X_i$ in graph $G$. Each conditional probability quantifies the dependence of each variable from its parents in the graph.

Given the independences represented by the graph $G$, it can be proved that the conditional probabilities uniquely determine a joint probability for the set of variables $X$ which can be obtained by means of the factorization theorem [?]:

$$P(X) = \prod_{i=0}^{n} P(X_i | \Pi(X_i))$$

B. Learning Bayesian network structures

Assume that we have a dataset $D = \{x^1, \ldots, x^m\}$ of observations of the variables in $X$ without missing values, i.e. each $x^i \in Val(X_1, \ldots, X_n)$ is a full set of values for the variables $X_1, \ldots, X_n$. Learning a Bayesian network consists in estimating a directed acyclic graph and the set of associated conditional distributions from dataset $D$. The induced network should be a good predictor for future observations of variables in $X$. The first part, estimating the graph structure is the essential one.

In practice, the most common approach is to define a score or metric which is a function $Score(G, D)$ depending of the graph and the data, which should measure the suitability of graph $G$ for explaining dataset $D$, but penalizing more complex networks at the same time. Once the metric is defined, the problem is to find the graph $G$ optimizing $Score(G, D)$ where $D$ is the available dataset of observations.

Different scoring metrics are used in order to evaluate the network structures. In general, there are local and global scores. A score is local, when it can be decomposed as a function of each node and its parents and therefore under local changes in the graph $G$ (for example adding an arc) the score has to be reevaluated only in the part of the graph involving these changes (the nodes in the arc). This property is very important for the efficiency of search optimization algorithms. Examples of local metrics are Bayesian scores as the K2 metric [?], the BDeu score [?], or other based of information theoretic concepts as the Bayesian Information Criterion (BIC) [?] or the Akaike Information Criterion (AIC) [?]. A global score depends of the full graph and local changes imply a full reevaluation of the score. The classification accuracy (CA) under 10-fold cross validation is an example of this metric. They are usually less efficient when searching in the full space of graphs, but they can be considered if the space is restricted to some specific types of graphs (for example when searching for trees).

C. Supervised learning with Bayesian networks

When learning a Bayesian network the general aim is to find a graph and conditional probability distributions which estimate the joint probability distribution of variables in $X$. However, in many cases we have a class variable of interest $C$ and what we want to estimate is the conditional probability distribution of $C$ given a set of predictor variables, $P(C = c | X_1 = x_1, \ldots, X_n = x_n)$, i.e. we have a supervised classification problem. Once this probability is estimated, under a set of observations $\{X_1 = x_1, \ldots, X_n = x_n\}$, and under 0-1 loss the predicted value $\hat{c}$ of $C$ is given by

$$\hat{c} = \arg \max \limits_{\{C\}} P(C = c | X_1 = x_1, \ldots, X_n = x_n)$$

Given that $P(C = c | X_1 = x_1, \ldots, X_n = x_n) \propto P(X_1 = x_1, ..., X_n = x_n | C = c) P(C = c)$, this value can also be computed as follows,

$$\hat{c} = \arg \max \limits_{\{C\}} P(X_1 = x_1, ..., X_n = x_n | C = c) P(C = c)$$

This problem could be approached by estimating a generic Bayesian network and then computing these probabilities on it, however in practice is more effective to restrict the class of estimated network structures, for example assuming that there is always a link from the class variable $C$ to each predictor $X_i$. Depending on the assumptions we can obtain different classifiers:

- Naive Bayes (NB) [?] assumes that each predictor variable is conditional independent of the rest of predictors given the class. It is the simplest one and the network structure is always the same. It is highly scalable, requiring a number of parameters linear in the number of features/predictors.
- To alleviate the conditional independence assumptions, some authors have proposed the Semi-naive Bayes (SNB) classifier [?]. The idea is to cluster some of the original variables in new multidimensional variables $A = (X_j)_{j \in J}$ where $J \subseteq \{1, \ldots, n\}$ and taking values on the Cartesian product $\times_{j \in J} Val(X_j)$.
- Another problem associated with the Naive Bayes classifier is that not always all the attributes are relevant for predicting the class. [?] proposed a semi-naive Bayes classifier in which some of the variables can be considered irrelevant by using a greedy wrapper approach.
• An alternative approach to tackle the conditional independence assumptions of NB classifier is to consider more complex structures: the Tree Augmented Naive Bayes (TAN) classifier [2] considers a tree for the dependences of predicting attributes. Though the structure depends of the data, the optimal tree can be efficiently constructed with a polynomial algorithm[?].
• Going an step beyond, we could consider that each attribute can have k parents instead of one as in TAN classifier. The k-dependence Bayesian (kDB) classifier [?] is the classifier obtained under this assumption, which uses more complex structures than TAN (if k ≥ 2).

D. Variable Mesh Optimization

Variable Mesh Optimization (VMO) [?] is a population metaheuristic where the solutions are distributed as a mesh in the M-dimensional space. This mesh contains P candidate solutions (n_1, n_2, ..., n_P), known in this context as nodes whose representation depends on the problem. Two operations: the expansion and contraction processes are carried out by VMO in order to explore the search space. These processes are used to introduce a suitable balance between exploitation and exploration in this algorithm. Both elements, mesh expansion and contraction, use a recommended minimum distance between the individuals ξ [?]. This distance, is reduced during the search and is responsible of keeping the diversity in the mesh. In following subsections each phase of the algorithm is described and the parameters are identified.

1) Mesh expansion operation: The expansion operation is carried out to move the population throughout the search space using different strategies to create new solutions. Each node created in the expansion process is introduced into a temporal mesh Mtemp that is initialized at the beginning of each iteration. The algorithm begins with an initial phase, where the initial mesh M is created with P solutions (parameter that identifies the number of nodes of the mesh to begin each iteration) randomly generated with uniform distribution. Each step of this operation is described as follows:

Phase 1.(Local exploration): In this step, the neighborhood of each node (n_i) in M is explored. First, the k (parameter that identifies the size of the neighborhood) closest nodes to n_i in current mesh M which are detected using the similarity function function for (n_i) are identified and the best node (fitness) in the neighborhood is selected as the local extreme (n^*_i). In this phase, only when n^*_i has better fitness than n_i a new node is generated using n_i and n^*_i through a function to combine solutions.

Phase 2.(Global exploration) This step explores in the direction of the global optimum (node in the current mesh with the best fitness, n_g). P − 1 new solutions are generated combining each node n_i in the mesh with n_g.

Phase 3.(Selective exploration) In this step, the interior frontier (more similar to the others nodes of the mesh, n_u) and the exterior frontier (less similar to the others nodes of the mesh, n_a) are identified using a similarity matrix. Then, both n_u and n_a are transformed to create two new nodes.

2) Mesh contraction process: The contraction operation is used to select the most representative nodes among M and Mtemp. These nodes will be maintained into M for the next iteration. The idea is to maintain nodes that are not too close to better ones, keeping a minimum distance between the nodes selected. Next an adaptative clearing operator is described in order to identify the most representative nodes:

Phase 4. All nodes (M ∪ Mtemp) are ordered depending on their fitness (first of the best fitness).

Phase 5. The similarity between each pair of nodes is calculated. If this value is greater than a threshold ξ, then the worst node is deleted. The ξ is calculated based on the problem and it changes depending of the iteration. This step reduces the number of nodes to B nodes.

Phase 6. The nodes with best fitness are selected as M for the next iteration. If M size is lower than P, then the mesh is completed with new random nodes.

In this process the nodes with better fitness and more diversity have a higher probability to be selected for the next iteration.

III. EXPERIMENTS

A. Learning using VMO algorithm

VMO algorithm can be applied to structural learning of a Bayesian network (BayesVMO) but some elements have to be adapted to the problem conditions. We implemented the VMO’s approach on Elvira system [?]. This software is an open source tool for working with Bayesian networks containing a variety of algorithms for learning and inference.

Each structure determines a classifier in the usual way: first, the conditional probability tables are also estimated from data. Then, the full Bayesian network is used as a classifier: the posterior of the class given the observations is computed, and the class value with highest posterior probability is chosen.

Representation: We represent the node of the mesh as a set of directed arcs (a, b), where a, b are variables of the BN and the order defines the direction of the arc (see Figure ?2). The root of each structure is represented by the class variable.

Fitness: Each structure is converted into a Bayesian Classifier using the data structures of Elvira. Later, the classification accuracy by cross validation over learning data is computed as fitness.
Calculate similarity($BN_1$, $BN_2$)

1) $d \leftarrow 0$
2) For each arc $a_{1,i} \in BN_1$:
   a) if $a_{1,i} \in BN_2$ then $d \leftarrow d + 1$
3) Return $d$

Fig. 2. Calculation of similarity between two BNs.

and the cardinality of the intersection set represents the similarity (see equation (4)) and code figure 3). This similarity is used in the phase 3 of the algorithm (Figure 2) to create a similarity matrix and find the BNs most (the sum of the similarities to other networks is greater) and least (the sum of the similarities to other networks is less) similar.

\[ \text{Similarity}_{n_1,n_2} = \text{Card}(n_1 \cap n_2) \] (4)

**Generation of BNs:** For local exploration, we used the similarity matrix to find the $k$ nearest neighbors of one structure on the current population. Then, Whenever a BN is created we have to check that there are no cycles. In case that a cycle is present, then one of its arcs is randomly selected and its direction is changed or removed (if the arc appears in both directions). This process is repeated until all cycles are eliminated. In order to generate a new BN, two operations are used:

- Random generation: Each arc is randomly generated and it is added to the BN if the two following constraints are true: the arc does not incorporate a cycle in the BN and it is not directed to the class variable. The process is repeated until a cycle is generated.
- Generation from two BNs: In order to generate a new BN from two BNs, two operations are used in our experiments. The union operation and union of the differences. The union operation obtains a new BN with all the arcs of the two operated BNs (see equation (5)). The union of the differences generates a new BN with the arcs present in one of the BNs but not in the other. (see equation (6)). The two operation are presented in figures 3 and 4.

\[ BN_{\text{new}} = (BN_1 \cup BN_2) \] (5)

\[ BN_{\text{new}} = (BN_1 \setminus BN_2) \cup (BN_2 \setminus BN_1) \] (6)

**Distance of clearing.** $\xi$ regulates the exploration and exploitation levels of the algorithm and for learning BNs is calculated according to the following equations:

\[ \xi = \begin{cases} 
0.8 \times A & \text{if } r < \frac{1}{4} \times C \\
0.7 \times A & \text{if } \frac{1}{4} \times C \leq r < \frac{2}{4} \times C \\
0.6 \times A & \text{if } \frac{2}{4} \times C \leq r < \frac{3}{4} \times C \\
0.5 \times A & \text{otherwise}
\end{cases} \] (7)

where $A$ represents the number of variables or features of the problem, $r$ the current iteration of the VMO algorithm and $C$ the number of iterations of the algorithm (stopping criterion).

**Selection strategy:** We used the clearing operation presented in the original VMO algorithm. In this case, the most representative BNs are selected. For this, the BNs are sorted by their fitness value. And then, every BN close to a better one is eliminated. The $\xi$ in the equation (7) is used in this process in order to obtain the more representative nodes. If the resulting number of nodes is less than the $P$ parameter then, new random nodes are generated in order to complete the initial mesh of next iteration.

Figure 4 shows the BayesVMO scheme. In order to obtain
Init Generate initial Mesh $M$ with $P$ BNs by uniform random. 
$M^{temp} \leftarrow M$. 
Identify current best BN $n_g \in M$.

Phase I For each $n_i$ in the mesh not local optimum:
- Obtain its neighbour with best fitness $n_i^*$. 
- $n_i^{ph1} = \text{combination}_1(n_i, n_i^*)$ using equation ??.
- $M^{temp} \leftarrow M^{temp} \cup n_i^{ph1}$.

Phase II For each $n_i$ in the mesh:
- $n_i^{ph2} = \text{combination}_2(n_i, n_g)$ using equation ??.
- $M^{temp} \leftarrow M^{temp} \cup n_i^{ph2}$.

Phase III Expand the mesh:
- To select $n_s, n_a \in M$ as the most and less similar to all.
- $n_i^{ph3} = \text{change}(n_s)$ and $n_i^{ph4} = \text{change}(n_a)$, randomly select a arc and change its direction. 
- $M^{temp} \leftarrow M^{temp} \cup n_i^{ph3} \cup n_i^{ph4}$.

Clearing Remove worst solutions from $M^{temp}$.
- Sort $M^{temp}$ by their fitness.
- For each $n_i, n_k \in M^{temp}$ calculate similarity $(\text{fitness}(n_i) \text{ better than fitness}(n_k))$ using equation ??.
- Remove $n_k$ if distance($n_i, n_k$) $\leq \xi_j$, where $\xi_j$ is defined by equation ??.

New $M$ Create next $M$ from $M^{temp}$.
- If $|M| < |M^{temp}|$ $M \leftarrow$ best solutions from $M^{temp}$.
- If $|M| > |M^{temp}|$ $M \leftarrow M^{temp} \cup$ random solutions.
- $M^{temp} \leftarrow M$.
- Identify current best solution $n_g \in M$.

End Iteration Return Phase I if not stopping criterion.

Fig. 5. VMO Scheme for learning of BNs.

the fitness value, the classification accuracy with 10-fold cross validation is used. We did not use such local metrics as, K2, BIC, MLD, or others type of score metrics.

B. Complexity

The BayesVMO algorithm complexity depends of two processes, the search and score methods. The search is applied by the VMO metaheuristic and the most complex operation is to obtain the similarity matrix of the population in each iteration of the algorithm. Its complexity is established as $O(CP^2A^2)$, where $C$ represents the number of algorithm iterations, $P$ the size of population and $A$ the number of attributes of the problem. $P^2$ represents the number the comparison on population (all against all) and $A^4$ the intersection operation between to structures ($A^2$ is the maximum number of arcs of a graph and to compute the similarity, each arc of one graph has to be searched for its presence in the other graph).

As scoring method, the classification accuracy by the operator of cross-validation is used and it is executed once by solution in each algorithm iteration. Moreover, the complexity of the score, also depends of a probabilistic propagation ($\Pr$) and size of data ($D$). Its complexity is established as $O(CPPrD)$ which increases the complexity of BayesVMO algorithm. The complexity of this algorithm is similar to the ByNet, BayesChaid and BayesPSO algorithms presented in the experiments section ?? (according to autor) and it is more complex than other simple Bayesian classifiers as BN K2, BN TAN, NB, SNB presented on the same section.

C. Benchmarks and parameters for experiments

Benchmark used for this experimental study is composed of 14 UCI ML repository data set [?] and the description of each one is shown in Table ??.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Discrete-continuous feature</th>
<th>Class</th>
<th>Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mammographic</td>
<td>4-1</td>
<td>2</td>
<td>961</td>
</tr>
<tr>
<td>Lung-cancer</td>
<td>56-0</td>
<td>3</td>
<td>32</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>13-6</td>
<td>2</td>
<td>155</td>
</tr>
<tr>
<td>E coli</td>
<td>0-7</td>
<td>8</td>
<td>336</td>
</tr>
<tr>
<td>Breast-cancer-w</td>
<td>10-0</td>
<td>2</td>
<td>683</td>
</tr>
<tr>
<td>Contact-lenses</td>
<td>4-0</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>Hayes-roth-m</td>
<td>4-0</td>
<td>2</td>
<td>132</td>
</tr>
<tr>
<td>Monk 1</td>
<td>7-0</td>
<td>2</td>
<td>432</td>
</tr>
<tr>
<td>Vote</td>
<td>16-0</td>
<td>2</td>
<td>300</td>
</tr>
<tr>
<td>Balance-scale</td>
<td>4-0</td>
<td>3</td>
<td>625</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>9-0</td>
<td>2</td>
<td>958</td>
</tr>
<tr>
<td>Iris</td>
<td>0-4</td>
<td>3</td>
<td>150</td>
</tr>
<tr>
<td>Labor</td>
<td>8-8</td>
<td>2</td>
<td>57</td>
</tr>
<tr>
<td>Soybean</td>
<td>35-0</td>
<td>2</td>
<td>47</td>
</tr>
</tbody>
</table>

The values of parameters used into the experimental studies are; size of the initial mesh $P = 12$, size of the neighborhood $k = 3$ and iteration numbers of the algorithm $C = 10000$. In this work we did not present any fine-tuning study for these parameters due to space limitation.

D. Results and comparisons with other algorithms

For the our comparative study, the results of PhD thesis [?] are used. In it, three bayesian classifiers are proposed by the autor and its results are compared with tradicional Bayesian classifiers (BN K2, BN TAN, NB, SNB). To continue, the three proposals are summarized:

- ByNet: Decision trees based on the CHAID technique are created and the Chi-square test is used in order to find the most significant variables related to the dependent variable.
• BayesChaid: This technique is similar to the previous one but the trees are created using the depth first search algorithm and breadth first search.

• BayesPSO: This model uses the Particle Swarm Optimization metaheuristic as search procedure and the classification accuracy as score.

The experiments have been carried out following the specifications of benchmark. Table ?? shows the average values (classification’s accuracy) for 25 independent runs of each algorithm using the parameter defined by the authors themselves. The results of the last three classifiers were carried out by the authors.

In [?] a framework for statistical analysis is presented in order to identify the best results. We have applied the post-hoc procedures [?] Iman-Davenport’s test [?] and Holm’s method as non parametric tests [?]. The Iman-Davenport’s test is similar to the Friedman’s test and is used to identify the significant differences in a group of algorithms (more than two). If detected differences are significant , then Holm’s test is applied to compare the control algorithm (the best ranking) with the remaining ones.

The results of Iman-Davenport’s test show that the hypothesis of equality has been rejected (statistical test value (3.929) is greater than the critical value (2.032)). Significant differences were detected in the group.

To continue, the Holm’s test is applied using the BayesVMO as control algorithm. Table ?? shows the results for this test, where for all cases the results archived by BayesVMO are significantly superior to all of the compared classifiers. This conclusion was reached because the p-values are smaller than the corresponding $\alpha/i$ values

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Z</th>
<th>p-values</th>
<th>$\alpha/i$</th>
<th>Hypothesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNB</td>
<td>4.243</td>
<td>2.19E-5</td>
<td>0.0062</td>
<td>Rejected</td>
</tr>
<tr>
<td>NB</td>
<td>3.864</td>
<td>1.11E-4</td>
<td>0.0071</td>
<td>Rejected</td>
</tr>
<tr>
<td>ByNet</td>
<td>3.795</td>
<td>1.47E-4</td>
<td>0.0083</td>
<td>Rejected</td>
</tr>
<tr>
<td>BN TAN</td>
<td>2.622</td>
<td>0.0087</td>
<td>0.01</td>
<td>Rejected</td>
</tr>
<tr>
<td>BN K2</td>
<td>2.415</td>
<td>0.0157</td>
<td>0.0166</td>
<td>Rejected</td>
</tr>
<tr>
<td>BayesChaid</td>
<td>2.242</td>
<td>0.0243</td>
<td>0.025</td>
<td>Rejected</td>
</tr>
<tr>
<td>BayesPSO</td>
<td>2.001</td>
<td>0.0453</td>
<td>0.05</td>
<td>Rejected</td>
</tr>
</tbody>
</table>

IV. Conclusion

In this work, we have adapted the Variable Mesh Optimization (VMO) to structural learning of Bayesian network (BayesVMO). We used as score metric for classification accuracy and adapted VMO’s operations to this problem. Then, we have carried out experiments using 14 UCI Repository data sets as benchmark. The results of BayesVMO have been compared with 8 classifiers of the state of art and the statistical analyses showed that the BayesVMO reaches the best results.

ACKNOWLEDGMENT

This research was supported by the FOCICYT project of the University State Technique of Quevedo and by the Spanish Ministry of Education and Science under project TIN2013-46638-C3-2-P and the European Regional Development Fund (FEDER).
### TABLE III
**Comparison between BayesVMO and others classifiers**

<table>
<thead>
<tr>
<th>DataSet</th>
<th>ByNet</th>
<th>BayesChaid</th>
<th>BayesPSO</th>
<th>BN K2</th>
<th>BN TAN</th>
<th>NB</th>
<th>SNB</th>
<th>BayesVMO</th>
</tr>
</thead>
<tbody>
<tr>
<td>mammographic</td>
<td>81.89</td>
<td>83.14</td>
<td>82.62</td>
<td>82.41</td>
<td>81.27</td>
<td>83.25</td>
<td>79.29</td>
<td>84.39</td>
</tr>
<tr>
<td>Lung-cancer</td>
<td>78.13</td>
<td>75.00</td>
<td>78.13</td>
<td>71.88</td>
<td>65.63</td>
<td>75.00</td>
<td>46.87</td>
<td>78.13</td>
</tr>
<tr>
<td>hepatitis</td>
<td>85.16</td>
<td>86.45</td>
<td>83.87</td>
<td>84.52</td>
<td>85.16</td>
<td>85.16</td>
<td>87.74</td>
<td>93.55</td>
</tr>
<tr>
<td>e colic</td>
<td>67.26</td>
<td>85.12</td>
<td>84.52</td>
<td>85.12</td>
<td>84.82</td>
<td>85.42</td>
<td>62.79</td>
<td>75.49</td>
</tr>
<tr>
<td>breast-cancer-w</td>
<td>90.78</td>
<td>97.51</td>
<td>97.36</td>
<td>97.36</td>
<td>95.31</td>
<td>97.36</td>
<td>97.83</td>
<td>97.85</td>
</tr>
<tr>
<td>contact-lenses</td>
<td>87.50</td>
<td>83.33</td>
<td>83.33</td>
<td>70.83</td>
<td>66.67</td>
<td>70.83</td>
<td>66.66</td>
<td>95.83</td>
</tr>
<tr>
<td>hayes-roth-m</td>
<td>57.58</td>
<td>81.06</td>
<td>74.24</td>
<td>72.73</td>
<td>67.42</td>
<td>80.30</td>
<td>57.57</td>
<td>84.09</td>
</tr>
<tr>
<td>monk 1</td>
<td>72.58</td>
<td>70.97</td>
<td>99.19</td>
<td>79.03</td>
<td>95.97</td>
<td>77.42</td>
<td>57.25</td>
<td>97.58</td>
</tr>
<tr>
<td>vote</td>
<td><strong>94.33</strong></td>
<td><strong>91.67</strong></td>
<td><strong>92.67</strong></td>
<td><strong>92.67</strong></td>
<td><strong>93.67</strong></td>
<td><strong>89.67</strong></td>
<td><strong>94.03</strong></td>
<td><strong>94.79</strong></td>
</tr>
<tr>
<td>Balance-scale</td>
<td>63.52</td>
<td>92.16</td>
<td><strong>93.92</strong></td>
<td><strong>92.16</strong></td>
<td><strong>92.96</strong></td>
<td><strong>92.16</strong></td>
<td><strong>82.56</strong></td>
<td><strong>86.72</strong></td>
</tr>
<tr>
<td>tic-tac-toe</td>
<td>70.25</td>
<td>72.96</td>
<td>72.05</td>
<td>76.62</td>
<td>76.83</td>
<td>69.62</td>
<td>73.06</td>
<td><strong>77.68</strong></td>
</tr>
<tr>
<td>iris</td>
<td>95.33</td>
<td>94.00</td>
<td>95.33</td>
<td>94.00</td>
<td>94.67</td>
<td>94.00</td>
<td>92.00</td>
<td><strong>98.00</strong></td>
</tr>
<tr>
<td>labor</td>
<td>84.21</td>
<td>87.72</td>
<td>89.47</td>
<td>91.22</td>
<td>89.47</td>
<td>89.47</td>
<td>90.00</td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>soybean</td>
<td>68.08</td>
<td>93.11</td>
<td>89.16</td>
<td>94.58</td>
<td>94.58</td>
<td>92.97</td>
<td>99.87</td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>average</td>
<td><strong>78.33</strong></td>
<td><strong>85.3</strong></td>
<td><strong>85.23</strong></td>
<td><strong>84.65</strong></td>
<td><strong>84.60</strong></td>
<td><strong>84.47</strong></td>
<td><strong>76.97</strong></td>
<td><strong>90.36</strong></td>
</tr>
</tbody>
</table>