Technical Report UPM-FI/DIA/2010-1

Multi-Dimensional Classification with Bayesian Networks

Concha Bielza MCBIELZA@FI.UPM.ES
Guangdi Li GUANGDI.LI@FI.UPM.ES
Pedro Larrañaga PEDRO.LARRANAGA@FI.UPM.ES
Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Madrid, Spain

Abstract

Multi-dimensional classification aims at finding a function that assigns a vector of class values to a given vector of features. In this paper, this problem is tackled by a general family of models, called multi-dimensional Bayesian network classifiers (MBCs). This probabilistic graphical model organizes class and feature variables as three different subgraphs: class subgraph, feature subgraph, and bridge (from class to features) subgraph. Under the standard 0-1 loss function, the most probable explanation (MPE) must be computed, for which we provide theoretical results in both general MBCs and in MBCs decomposable into maximal connected components. Moreover, when computing the MPE, the vector of class values is covered by following a special ordering (gray code). Under other loss functions defined in accordance with a decomposable structure, we derive theoretical results on how to minimize the expected loss. Besides these inference issues, the paper presents flexible algorithms for learning MBC structures from data based on filter, wrapper and hybrid approaches. The cardinality of the search space is also given. New performance evaluation metrics adapted from the single-class setting are introduced. Experimental results with three benchmark data sets are encouraging, and they outperform state-of-the-art algorithms for multi-label classification.

Keywords: Multi-Dimensional Outputs, Bayesian Network Classifiers, Learning from Data, MPE, Multi-Label Classification

1. Introduction

In this paper we are interested in classification problems where there are multiple class variables $C_1, ..., C_d$. Therefore the *multi-dimensional classification* problem consists of finding a function h that assigns to each instance given by a vector of m features $\mathbf{x} = (x_1, ..., x_m)$ a vector of d class values $\mathbf{c} = (c_1, ..., c_d)$:

$$h : \Omega_{X_1} \times \cdots \times \Omega_{X_m} \rightarrow \Omega_{C_1} \times \cdots \times \Omega_{C_d}$$

$$(x_1, ..., x_m) \mapsto (c_1, ..., c_d)$$

We assume that C_i is a discrete variable, for all $i = 1, ..., d$, with Ω_{C_i} denoting its sample space and $\mathcal{I} = \Omega_{C_1} \times \cdots \times \Omega_{C_d}$, the space of joint configurations of the class variables. Analogously, Ω_{X_j} is the sample space of the discrete feature variable X_j, for all $j = 1, ..., m$.

Many application domains include multi-dimensional classification problems: a text document or a semantic scene can be assigned to multiple topics, a gene can have multiple biological functions, a patient may suffer from multiple diseases, a patient may become resistant to multiple drugs for HIV treatment, a physical device can break down due to multiple components failing, etc.

Multi-dimensional classification is a more difficult problem than the single-class case. The main problem is that there is a large number of possible class label combinations, $|\mathcal{I}|$, and a corresponding sparseness of available data. In a typical scenario where an instance $\mathbf{x}$ is assigned to the most likely combination of classes (0-1 loss function), the aim is to compute $\arg\max_{c_1,...,c_d} p(C_1 = c_1, ..., C_d = c_d|\mathbf{x})$. It holds that $p(C_1 = c_1, ..., C_d = c_d|\mathbf{x}) \propto p(C_1 = c_1, ..., C_d = c_d, \mathbf{x})$, which requires $|\mathcal{I}| \cdot |\Omega_{X_1} \times \cdots \times \Omega_{X_m}|$ parameters to be assigned. In the single-class case, $|\mathcal{I}|$ is just $|\Omega_C|$ rather than $|\Omega_{C_1} \times \cdots \times \Omega_{C_d}|$. Thus, if all variables are binary, we need 2^{d+m} parameters, whereas with one class variable, 2^{1+m}. The factorization of this joint probability distribution when using a Bayesian network (BN) can somehow reduce the number of parameters required and will be our starting point.

Standard (one-class) BN classifiers cannot be straightforwardly applied to this multi-dimensional setting. On the one hand, the problem could be transformed into a single-class problem if a compound class variable modeling all possible combinations of classes is constructed. However, this class variable would have too many values, and even worse, the model would not capture the structure of the classification problem (dependencies among class variables and also among class variables and features). On the other hand, we could approach the multi-dimensional problem by constructing one independent classifier for each class variable. However, this would not capture the interactions among class variables, and the most likely class label for each independent classifier –marginal classifications–, after being assembled as a d-dimensional vector, might not coincide with the most likely vector of class labels of the observed data.

As we will show below, the few proposals found in the literature on multi-dimensional BN classifiers (MBCs) are limited. In this paper, we propose a comprehensive theory of MBCs, including their extended definition, learning from data algorithms that cover all the possibilities (wrapper, filter and hybrid score+search strategies), and results on how to perform total abduction for the exact inference of the most probable explanation (MPE). MPE computation is the main aim in 0-1 loss function classification problems but involves a high computational cost in the multi-dimensional setting. Several contributions are designed here to reduce this computational load: the introduction of special decomposed MBCs, their extension to non 0-1 loss function problems that respect this decomposition, and a particular and favorable way of enumerating all the $(c_1, ..., c_d)$ configurations instead of using a brute-force approach.

The paper is organized as follows. Section 2 defines MBCs. Section 3 covers different contributions for the MPE computation and introduces a restricted structure of decomposable MBCs where MPE is easier to compute. Section 4 extends these ideas to compute the Bayes decision rule with certain loss functions that we call additive CB-decomposable loss functions. Section 5 presents performance measures suitable for evaluating MBCs. Section 6 describes wrapper, filter and hybrid algorithms to learn MBCs from data. It also provides the cardinality of the MBC structure space where these algorithms search for. Section 7 contains experimental results with three benchmark data sets. Section 8 reviews the work related to multi-dimensional classification, with special emphasis on papers using (simpler) MBCs. Finally, Section 9 sums up the paper with some conclusions.

2. Multi-Dimensional Bayesian Network Classifiers

A Bayesian network over a finite set $\mathcal{V} = \{Z_1, ..., Z_n\}$, $n \geq 1$, of discrete random variables is a pair $B = (\mathcal{G}, \Theta)$, where $\mathcal{G}$ is an acyclic directed graph whose vertices correspond to the random variables and Θ is a set of parameters $\theta_{z|\mathbf{pa}(z)} = p(z|\mathbf{pa}(z))$, where $\mathbf{pa}(z)$ is a value of the set of variables $\mathbf{Pa}(Z)$, parents of the Z variable in the graphical structure $\mathcal{G}$ (Pearl, 1988; Koller and Friedman, 2009). B defines a joint probability distribution p_B over $\mathcal{V}$ given by

$$p_B(z_1, ..., z_n) = \prod_{i=1}^{n} p(z_i|\mathbf{pa}(z_i)). \tag{1}$$

A *multi-dimensional Bayesian network classifier* is a Bayesian network specially designed to solve classification problems including multiple class variables in which instances described by a number of features have to be assigned to a combination of classes.

Definition 1 ***(Multi-dimensional Bayesian network classifier)*** *In an* MBC *denoted by* $B = (\mathcal{G}, \Theta)$*, the graph* $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ *has the set* $\mathcal{V}$ *of vertices partitioned into two sets* $\mathcal{V}_C = \{C_1, ..., C_d\}$*,* $d \geq 1$*, of class variables and* $\mathcal{V}_X = \{X_1, ..., X_m\}$*,* $m \geq 1$*, of feature variables* $(d + m = n)$*.* $\mathcal{G}$ *also has the set* $\mathcal{A}$ *of arcs partitioned into three sets,* $\mathcal{A}_C$*,* $\mathcal{A}_X$*,* $\mathcal{A}_{CX}$*, such that:*

- $\mathcal{A}_C \subseteq \mathcal{V}_C \times \mathcal{V}_C$ *is composed of the arcs between the class variables having a subgraph* $\mathcal{G}_C = (\mathcal{V}_C, \mathcal{A}_C)$ –class subgraph– *of* $\mathcal{G}$ *induced by* $\mathcal{V}_C$*.*
- $\mathcal{A}_X \subseteq \mathcal{V}_X \times \mathcal{V}_X$ *is composed of the arcs between the feature variables having a subgraph* $\mathcal{G}_X = (\mathcal{V}_X, \mathcal{A}_X)$ –feature subgraph– *of* $\mathcal{G}$ *induced by* $\mathcal{V}_X$*.*
- $\mathcal{A}_{CX} \subseteq \mathcal{V}_C \times \mathcal{V}_X$ *is composed of the arcs from the class variables to the feature variables having a subgraph* $\mathcal{G}_{CX} = (\mathcal{V}, \mathcal{A}_{CX})$ –bridge subgraph– *of* $\mathcal{G}$ *connecting class and feature variables.*

This definition extends that in van der Gaag and de Waal (2006), which requires two additional conditions (see Section 6.4). Figure 1 shows an example of an MBC structure and its different subgraphs.

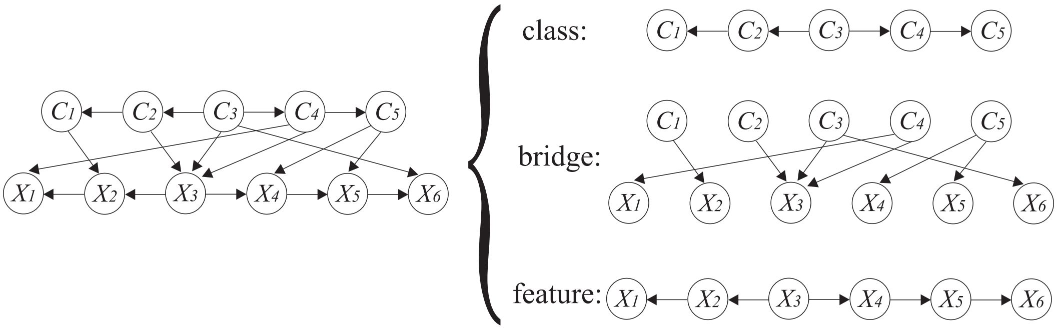


Figure 1: An example of an MBC structure with its three subgraphs

Note that different graphical structures for the class and feature subgraphs may give rise to different families of MBCs. In general, class and feature subgraphs may be: empty, directed trees, forest of trees, polytrees, and general directed acyclic graphs (DAG). The different families of MBCs will be denoted as `class subgraph structure-feature subgraph structure` MBC, where the possible structures are the above five. Thus, if both the class and feature subgraphs are directed trees, then this subfamily is a `tree-tree` MBC. Other examples are shown in Figure 2.

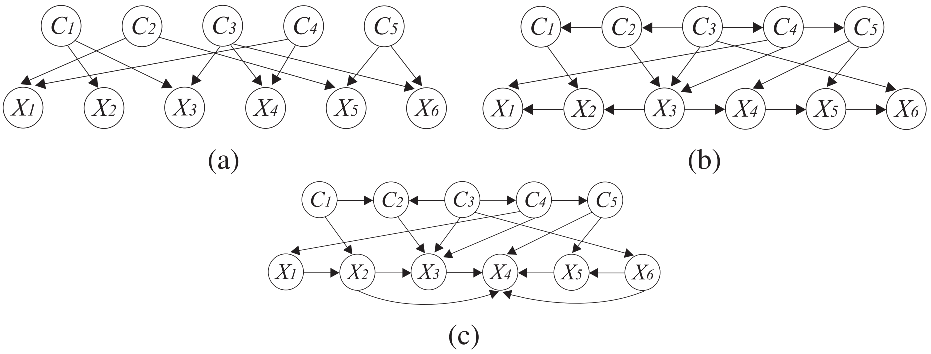


Figure 2: Examples of structures belonging to different families of MBCs. (a) `Empty-empty` MBC; (b) `Tree-tree` MBC; (c) `Polytree-DAG` MBC

Note that the well-known Bayesian classifiers: naïve Bayes (Minsky, 1961), selective naïve Bayes (Langley and Sage, 1999), tree-augmented naïve Bayes (Friedman et al., 1997), selective tree-augmented naïve Bayes (Blanco et al., 2005) and k-dependence Bayesian classifiers (Sahami, 1996) are special cases of MBCs where $d = 1$. Several MBC structures have been used in the literature: `tree-tree` MBC (van der Gaag and de Waal, 2006), `polytree-polytree` MBC (de Waal and van der Gaag, 2007) and a special `DAG-DAG` MBC (Rodríguez and Lozano, 2008).

The following theorem extends the well-known result that states that given a 0-1 loss function in a (one-dimensional) classification problem, the Bayes decision rule is to select the class label that maximizes the posterior probability of the class variable given the features. This supports the use of the percentage of correctly classified instances (or classifier accuracy) as a performance measure. In Section 5 we extend the definition of accuracy to our multi-dimensional setting.

Theorem 2 *Let $\lambda(\boldsymbol{c}'_i, \boldsymbol{c}_j)$ be a 0-1 loss function that assigns a unit loss to any error, i.e. whenever $\boldsymbol{c}'_i \neq \boldsymbol{c}_j$, where $\boldsymbol{c}'_i$ is the d-dimensional vector of class values output by a model and $\boldsymbol{c}_j$ contains the true class value and assigns no loss to a correct classification, i.e. when $\boldsymbol{c}'_i = \boldsymbol{c}_j$.*

Let $R(\boldsymbol{c}'_i|\boldsymbol{x}) = \sum_{j=1}^{|\mathcal{I}|} \lambda(\boldsymbol{c}'_i, \boldsymbol{c}_j)p(\boldsymbol{c}_j|\boldsymbol{x})$ be the expected loss or conditional risk, where $\boldsymbol{x} = (x_1, ..., x_m)$ is a vector of feature values and $p(\boldsymbol{c}_j|\boldsymbol{x})$ is the joint posterior probability, provided by a model, of the vector of the class value $\boldsymbol{c}_j$ given the observation $\boldsymbol{x}$.

Then the Bayes decision rule that minimizes the expected loss $R(\boldsymbol{c}'_i|\boldsymbol{x})$ is equivalent to selecting the i that maximizes the posterior probability $p(\boldsymbol{c}_i|\boldsymbol{x})$, that is,

$$\min_{\boldsymbol{c}'_i \in \mathcal{I}} R(\boldsymbol{c}'_i|\boldsymbol{x}) \Leftrightarrow \max_{\boldsymbol{c}_i \in \mathcal{I}} p(\boldsymbol{c}_i|\boldsymbol{x})$$

Proof. The proof is straightforward and analogous to the single-class variable case (Duda et al., 2000) using $\mathbf{C} = (C_1, ..., C_d)$ as the (d-dimensional) class variable, i.e.

$$R(\mathbf{c}'_i|\mathbf{x}) = \sum_{j=1}^{|\mathcal{I}|} \lambda(\mathbf{c}'_i, \mathbf{c}_j) p(\mathbf{c}_j|\mathbf{x}) = \sum_{j \neq i} p(\mathbf{c}_j|\mathbf{x}) = 1 - p(\mathbf{c}_i|\mathbf{x}).$$

■

Therefore, the multi-dimensional classification problem with a 0-1 loss is equivalent to computing a type of maximum a posteriori (MAP), known as *most probable explanation* (MPE), also called total abduction (Pearl, 1988). This has been shown to be a NP-hard problem for Bayesian networks (Shimony, 1994). Approximating the MPE problem is also NP-hard (Abdelbar and Hedetniemi, 1998).

However, the special structure that defines the MBC will alleviate somewhat MPE computation under certain circumstances, as shown in the next section.

3. Theoretical Results on MPE

This section presents several results concerning MPE computation for MBCs. First, computation of the joint posterior distribution on the d class variables when computing the MPE in the multi-dimensional setting appears to be a more difficult problem than in the single-class case. Second, this computation is alleviated by covering all the joint configurations in a special order (gray code). An upper bound for the savings achieved with respect to a brute-force approach is provided. Finally, when the graph union of class and bridge subgraphs of an MBC structure is decomposed into a number, r, of connected subgraphs, the maximization problem for computing the MPE can be transformed into r maximization problems operating in lower dimensional spaces. In this case of what we call class-bridge decomposable MBCs, we provide results about MPE computation and the savings using a variant of gray codes.

The NP-hardness of MPE computation in general Bayesian networks has led to the design of both exact and approximate algorithms. Exact algorithms include approaches using junction trees (Dawid, 1992), variable elimination (Li and D'Ambrosio, 1993; Dechter, 1999), and branch-and-bound search (Kask and Dechter, 2001; Marinescu and Dechter, 2009). Approximate algorithms cover the use of genetic algorithms (Gelsema, 1995; Rojas-Guzmán and Kramer, 1993), taboo search, hill climbing and sequential initialization (Park and Darwiche, 2004), stochastic local search algorithms (Kask and Dechter, 1999; Hutter et al., 2005), the so-called mini-bucket approach based on variable elimination (Dechter and Rish, 1997), best-first search (Shimony and Charniak, 1990) and linear programming (Santos, 1991).

MPE computation is even worse in the case of MBCs, since having d class variables increases the number of possible configurations exponentially, i.e. given evidence $\mathbf{x}$ we have to get

$$\mathbf{c}^* = (c_1^*, ..., c_d^*) = \arg \max_{c_1,...,c_d} p(C_1 = c_1, ..., C_d = c_d|\mathbf{x}). \tag{2}$$

De Waal and van der Gaag (2007) show that the classification problem can be solved in polynomial time if the feature subgraph has bounded treewidth and the number of class variables is restricted (see their Theorem 1). This implies that the connectivity of the class subgraph is irrelevant for the feasibility of classification.

Thanks to the specific structure of MBCs and a special way of moving within the $\mathcal{I}$ space of joint configurations of the class variables, we will be able, despite this high complexity, to reduce the computations performed to obtain the posterior probability $p(C_1 = c_1, ..., C_d = c_d|\mathbf{x})$ and finally get the MPE.

The main motivation lies in the similarity between the posterior probability of two configurations that have the same class values in all components but one.

Example 1: Given the MBC structure of Figure 3, where all variables are assumed to be binary (0/1) and $\mathbf{x} = (x_1, x_2, x_3, x_4)$, the posterior probabilities of configurations $(0, 0, 0)$ and $(1, 0, 0)$ for (C_1, C_2, C_3) satisfy:

$$\begin{aligned}
\frac{p((0,0,0)|\mathbf{x})}{p((1,0,0)|\mathbf{x})} &= \frac{p(0,0,0,\mathbf{x})}{p(1,0,0,\mathbf{x})} \\
&= \frac{p(C_1 = 0|C_2 = 0)p(C_2 = 0)p(C_3 = 0|C_2 = 0)p(X_1 = x_1|C_1 = 0, C_2 = 0)}{p(C_1 = 1|C_2 = 0)p(C_2 = 0)p(C_3 = 0|C_2 = 0)p(X_1 = x_1|C_1 = 1, C_2 = 0)} \cdot \\
&\quad \frac{p(X_2 = x_2|C_1 = 0, C_2 = 0, C_3 = 0)p(X_3 = x_3|C_3 = 0, X_1 = x_1)}{p(X_2 = x_2|C_1 = 1, C_2 = 0, C_3 = 0)p(X_3 = x_3|C_3 = 0, X_1 = x_1)} \cdot \\
&\quad \frac{p(X_4 = x_4|C_2 = 0, C_3 = 0, X_1 = x_1)}{p(X_4 = x_4|C_2 = 0, C_3 = 0, X_1 = x_1)} \\
&= \frac{p(C_1 = 0|C_2 = 0)p(X_1 = x_1|C_1 = 0, C_2 = 0)p(X_2 = x_2|C_1 = 0, C_2 = 0, C_3 = 0)}{p(C_1 = 1|C_2 = 0)p(X_1 = x_1|C_1 = 1, C_2 = 0)p(X_2 = x_2|C_1 = 1, C_2 = 0, C_3 = 0)}
\end{aligned}$$

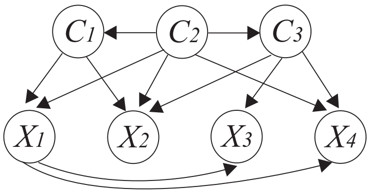


Figure 3: An example of MBC structure

This may be generalized in the following proposition.

Proposition 3 *Given an MBC and an instantiation of all the feature variables $\boldsymbol{x} = (x_1, ..., x_m)$, then the ratio of posterior distributions of two d-dimensional class configurations $\boldsymbol{c} = (c_1, ..., c_d)$ and $\boldsymbol{c}' = (c'_1, ..., c'_d)$ is given by*

$$\frac{p(\boldsymbol{c}|\boldsymbol{x})}{p(\boldsymbol{c}'|\boldsymbol{x})} = \frac{\prod_{C_i \in \mathcal{W}} p(C_i = c_i|\boldsymbol{pa}(c_i))}{\prod_{C_i \in \mathcal{W}} p(C_i = c'_i|\boldsymbol{pa}(c'_i))} \cdot \frac{\prod_{X_j \in Ch(\mathcal{W})} p(X_j = x_j|\boldsymbol{pa}(x_j))}{\prod_{X_j \in Ch(\mathcal{W})} p(X_j = x_j|\boldsymbol{pa}'(x_j))},$$

where $\mathcal{W} = \{C_i \in \mathcal{V}_C \,|\, \exists l \in \{1, ..., d\}, c_l \neq c'_l\}$, $\boldsymbol{pa}'(x_j)$ denotes the configuration of $\boldsymbol{Pa}(X_j)$ compatible with $\boldsymbol{x}$ and $\boldsymbol{c}'$, and $Ch(\mathcal{W})$ denotes the feature variables that are children of variables in set $\mathcal{W}$.

Proof: In the numerator,

$$p(\mathbf{c}|\mathbf{x}) = \frac{1}{p(\mathbf{x})} \prod_{C_i \in \mathcal{W}} p(C_i = c_i|\mathbf{pa}(c_i)) \prod_{C_i \notin \mathcal{W}} p(C_i = c_i|\mathbf{pa}(c_i))$$
$$\prod_{X_j \in Ch(\mathcal{W})} p(X_j = x_j|\mathbf{pa}(x_j)) \prod_{X_j \notin Ch(\mathcal{W})} p(X_j = x_j|\mathbf{pa}(x_j))$$

In the denominator, the factorization for $p(\mathbf{c}'|\mathbf{x})$ is analogous and its first, third, and fifth factors coincide with those in $p(\mathbf{c}|\mathbf{x})$. This leads directly to the final result. ■

Corollary 4 *In the above situation, where* **c** *and* **c**′ *now differ by only one component l, i.e.* $c_i = c'_i$ $\forall i \neq l$ *and* $c_l \neq c'_l$*, then*

$$\frac{p(\boldsymbol{c}|\boldsymbol{x})}{p(\boldsymbol{c}'|\boldsymbol{x})} = \frac{p(C_l = c_l|\boldsymbol{pa}(c_l))}{p(C_l = c'_l|\boldsymbol{pa}(c'_l))} \cdot \frac{\prod_{X_j \in Ch(C_l)} p(X_j = x_j|\boldsymbol{pa}(x_j))}{\prod_{X_j \in Ch(C_l)} p(X_j = x_j|\boldsymbol{pa}'(x_j))}.$$

These configurations **c** and **c**′ differing by one component (as in Example 1) provide more savings than in the general case of Proposition 3. For simplicity's sake, we can choose c_l and c_l' such that $|c_l - c'_l| = 1$. In this case, an adaptation of the gray code introduced by Guan (1998) is proposed for enumerating all the $(c_1, ..., c_d)$ configurations in a special order. Guan's $(n; k)$-gray code is a special sequence enumerating all elements in $(Z_n)^k$, that is, vectors of k components each taking values in the space $\{0, 1, ..., n - 1\}$. Therefore, components are restricted to being in the same range. We, however, extend gray codes to different ranges r_i, $i = 1, ..., d$, having $(r_1, ..., r_d; d)$-gray codes.

Definition 5 *(*$(r_1, ..., r_d; d)$***-gray code)*** *Given a vector* $(C_1, ..., C_d)$ *with each component* C_i *taking values in* $\{0, 1, ..., r_i - 1\}$*,* $i = 1, ..., d$*, an* $(r_1, ..., r_d; d)$-gray code *is a sequence that enumerates all the configurations* $(c_1, ..., c_d)$ *such that each pair of adjacent configurations differs by only one component and the difference is either 1 or -1.*

Example 2: Figure 4 shows the sequence of configurations for a $(3, 3, 2; 3)$-gray code, i.e. triplets where the first component takes values in $\{0, 1, 2\}$, the second in $\{0, 1, 2\}$ and the third in $\{0, 1\}$.

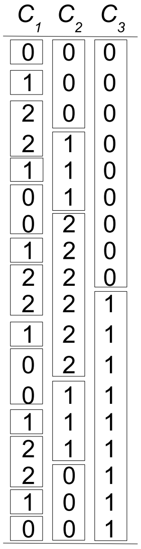
In this example, if S_i denotes the number of changes in the ith component to cover the whole gray code, then $S_3 = 1, S_2 = 4, S_1 = 12$ (see the boxes in Figure 4).

The general formula for S_i follows.

Proposition 6 *In the* $(r_1, ..., r_d; d)$*-gray code, the number of changes,* S_i*, in the ith component, is given by*

$$S_i = \begin{cases} \prod_{j=i}^{d} r_j - \prod_{j=i+1}^{d} r_j, & 1 \leq i \leq d - 1 \\ r_d - 1, & i = d \end{cases}$$

Moreover, $\sum_{i=1}^{d} S_i = \prod_{j=1}^{d} r_j - 1.$



C_1	C_2	C_3
0	0	0
1	0	0
2	0	0
2	1	0
1	1	0
0	1	0
0	2	0
1	2	0
2	2	0
2	2	1
1	2	1
0	2	1
0	1	1
1	1	1
2	1	1
2	0	1
1	0	1
0	0	1

Figure 4: (3, 3, 2; 3)-gray code

Proof: For $i = d, S_d = r_d - 1$ since the last component is the one is moved least by gray coding. For $1 \leq i \leq d-1$, $S_i = r_d \cdot r_{d-1} \cdots r_{i+1} \cdot (r_i - 1)$, since the ith component has $r_i - 1$ changes for each partial configuration of fixed values in components $i+1, ..., d$, which are $r_d \cdot r_{d-1} \cdots r_{i+1}$.

Also,

$$\sum_{i=1}^{d} S_i = \sum_{i=1}^{d-1} (\prod_{j=i}^{d} r_j - \prod_{j=i+1}^{d} r_j) + r_d - 1$$

$$= (r_1 \cdots r_d) - (r_2 \cdots r_d) + (r_2 \cdots r_d) - (r_3 \cdots r_d) + \cdots + (r_{d-1} \cdot r_d) - r_d + r_d - 1 = r_1 \cdots r_d - 1.$$

■

Computations to obtain MPE in MBCs are reduced by using these gray codes. The next theorem shows the savings and an upper bound when comparing the number of factors needed in the posterior probability computations with gray codes, F_{GC}, and with brute-force, F_{BF}.

Theorem 7 *Given an MBC with m feature variables and d class variables, where $\mathcal{I}$ is the space of joint configurations of the class variables, then the number of factors needed in the posterior probability computations with gray codes, F_{GC}, and with brute-force, F_{BF}, satisfy:*

(i) $F_{GC} = m + d + \sum_{i=1}^{d} S_i H_i$

(ii) $\frac{F_{GC}}{F_{BF}} < \frac{1}{|\mathcal{I}|} + \frac{H_{Max}}{m+d}$,

where $H_i = 1 + h_i$, h_i being the number of children (in the MBC) of the class variable that changes the ith in the gray code, and $H_{Max} = \max_{1 \leq i \leq d} H_i$.

Proof:

(i) $F_{GC} = m + d + \sum_{i=1}^{d} S_i H_i$, since $m + d$ corresponds to the number of factors (without savings) for calculating the posterior probability for the first configuration where the gray code starts from, and, using Corollary 4, each configuration that changes its ith class variable in the gray code requires $1 + h_i = H_i$ new factors. Taking into account that the number of changes in the ith class variable along the gray code sequence is S_i, the second term, $\sum_{i=1}^{d} S_i H_i$, gives the total number of factors required by all the configurations except the first one.

(ii) Obviously, $F_{BF} = (m + d)|\mathcal{I}|$. Also,

$$F_{GC} = m + d + \sum_{i=1}^{d} S_i H_i \leq m + d + H_{Max} \sum_{i=1}^{d} S_i = m + d + H_{Max}(|\mathcal{I}| - 1),$$

since $\sum_{i=1}^{d} S_i = |\mathcal{I}| - 1$ is the total number of changes in the gray code. Therefore,

$$\frac{F_{GC}}{F_{BF}} \leq \frac{m + d + H_{Max}(|\mathcal{I}| - 1)}{(m + d)|\mathcal{I}|} < \frac{1}{|\mathcal{I}|} + \frac{H_{Max}}{m + d}.$$

■

Example 1 (continued): Given the MBC structure of Figure 3, where variables C_1 and C_2 now take three possible values and C_3 is still binary, we have that $d = 3, m = 4, H_{Max} = 4, |\mathcal{I}| = 18$.
Thus,

$$\frac{F_{GC}}{F_{BF}} = \frac{63}{126},$$

and the upper bound is $\frac{F_{GC}}{F_{BF}} < \frac{79}{126}$.

Therefore, with gray codes the number of factors is reduced by half (63 against 126), while the upper bound is a little bit higher.

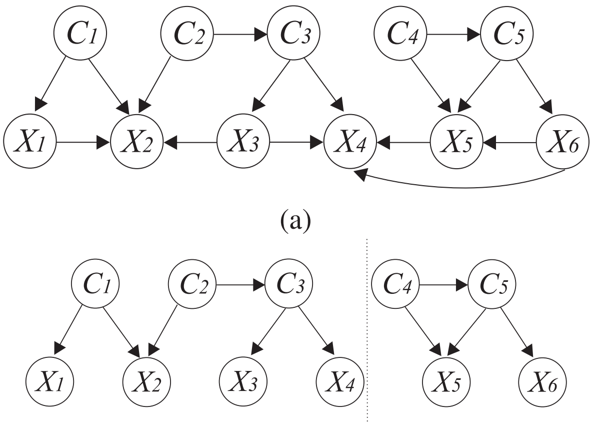
Definition 8 ***(CB-decomposable MBC)*** *Suppose we have an MBC where $\mathcal{G}_C$ and $\mathcal{G}_{CX}$ are its associated class and bridge subgraphs respectively. We say that the MBC is* class-bridge decomposable *(CB-decomposable for short) if:*

1. *$\mathcal{G}_C \cup \mathcal{G}_{CX}$ can be decomposed as $\mathcal{G}_C \cup \mathcal{G}_{CX} = \bigcup_{i=1}^{r}(\mathcal{G}_{Ci} \cup \mathcal{G}_{(CX)_i})$, where $\mathcal{G}_{Ci} \cup \mathcal{G}_{(CX)_i}$, with $i = 1, ..., r$, are its r maximal connected components*[1]*, and*

2. *$Ch(\mathcal{V}_{Ci}) \cap Ch(\mathcal{V}_{Cj}) = \emptyset$, with $i, j = 1, ..., r$ and $i \neq j$, where $Ch(\mathcal{V}_{Ci})$ denotes the children of all the variables in $\mathcal{V}_{Ci}$, the subset of class variables in $\mathcal{G}_{Ci}$ (*non-shared children property*).*

Example 3: Let us take the MBC structure shown in Figure 5(a). It is CB-decomposable with $r = 2$, as shown in Figure 5(b). The subgraph to the left of the dashed vertical line is $\mathcal{G}_{C1} \cup \mathcal{G}_{(CX)_1}$, i.e. the first maximal connected component. Analogously, $\mathcal{G}_{C2} \cup \mathcal{G}_{(CX)_2}$ to the right-hand side is the second maximal connected component. That is, $\mathcal{V}_{C1} = \{C_1, C_2, C_3\}, \mathcal{V}_{C2} = \{C_4, C_5\}, Ch(\mathcal{V}_{C1}) = \{X_1, X_2, X_3, X_4\}$ and $Ch(\mathcal{V}_{C2}) = \{X_5, X_6\}$. Note that $Ch(\{C_1, C_2, C_3\}) \cap Ch(\{C_4, C_5\}) = \emptyset$ as required.

[1] A graph is said to be connected if there is a path between every pair of vertices in its undirected version.



(a)

(b)

Figure 5: (a) A CB-decomposable MBC. (b) Its two maximal connected components

Theorem 9 *Given a CB-decomposable MBC where $\mathcal{I}_i = \prod_{C\in\mathcal{V}_{Ci}} \Omega_C$ represents the sample space associated with $\mathcal{V}_{Ci}$, then*

$$\begin{aligned} &\max_{c_1,...,c_d} p(C_1 = c_1, ..., C_d = c_d | X_1 = x_1, ..., X_m = x_m) \\ &\propto \prod_{i=1}^{r} \max_{\mathbf{c}^{\downarrow \mathcal{V}_{Ci}} \in \mathcal{I}_i} \prod_{C\in\mathcal{V}_{Ci}} p(c|\boldsymbol{pa}(c)) \prod_{X\in Ch(\mathcal{V}_{Ci})} p(x|\boldsymbol{pa}_{\mathcal{V}_C}(x), \boldsymbol{pa}_{\mathcal{V}_X}(x)), \end{aligned} \tag{3}$$

where $\mathbf{c}^{\downarrow \mathcal{V}_{Ci}}$ represents the projection of vector $\mathbf{c}$ to the coordinates found in $\mathcal{V}_{Ci}$.

Proof. By using firstly the factorization given in (1) and then the grouping of all the variables according to the CB-decomposable MBC assumption, we have that

$$\begin{aligned} &p(C_1 = c_1, ..., C_d = c_d | X_1 = x_1, ..., X_m = x_m) \\ &\propto \prod_{C\in\mathcal{V}_C} p(c|\mathbf{pa}(c)) \prod_{X\in\mathcal{V}_X} p(x|\mathbf{pa}_{\mathcal{V}_C}(x), \mathbf{pa}_{\mathcal{V}_X}(x)) \\ &= \prod_{i=1}^{r} \prod_{C\in\mathcal{V}_{Ci}} p(c|\mathbf{pa}(c)) \prod_{X\in Ch(\mathcal{V}_{Ci})} p(x|\mathbf{pa}_{\mathcal{V}_C}(x), \mathbf{pa}_{\mathcal{V}_X}(x)). \end{aligned}$$

Maximizing the last expression with respect to all the class variables amounts to maximizing over the identified class variables of the maximal connected components. The new maximization problems are carried out on lower dimensional subspaces than originally, thereby reducing the computational cost. Note that the feature subgraph structure is irrelevant in this process. ■

Given $\mathbf{x}$, each expression to be maximized in Equation (3) will be denoted as $\phi_i^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{Ci}})$, i.e.

$$\phi_i^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{Ci}}) = \prod_{C\in\mathcal{V}_{Ci}} p(c|\mathbf{pa}(c)) \cdot \prod_{X\in Ch(\mathcal{V}_{Ci})} p(x|\mathbf{pa}_{\mathcal{V}_C}(x), \mathbf{pa}_{\mathcal{V}_X}(x))$$

It holds that $\phi_i^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \propto p(C^{\downarrow \mathcal{V}_{C_i}} = \mathbf{c}^{\downarrow \mathcal{V}_{C_i}} | \mathbf{x})$.

Example 3 (continued): For the CB-decomposable MBC in Figure 5(a), we have that

$$\begin{aligned}
& \max_{c_1,...,c_5} p(C_1 = c_1, ..., C_5 = c_5 | X_1 = x_1, ..., X_6 = x_6) \\
& \propto \max_{c_1,...,c_5} p(c_1)p(c_2)p(c_3|c_2)p(c_4)p(c_5|c_4)p(x_1|c_1)p(x_2|c_1, c_2, x_1, x_3) \\
& \cdot \; p(x_3|c_3)p(x_4|c_3, x_3, x_5, x_6)p(x_5|c_4, c_5, x_6)p(x_6|c_5) \\
& = \left[\max_{c_1,c_2,c_3} p(c_1)p(c_2)p(c_3|c_2)p(x_1|c_1)p(x_2|c_1, c_2, x_1, x_3)p(x_3|c_3)p(x_4|c_3, x_3, x_5, x_6) \right] \\
& \cdot \left[\max_{c_4,c_5} p(c_4)p(c_5|c_4)p(x_5|c_4, c_5, x_6)p(x_6|c_5) \right] \\
& = \left[\max_{c_1,c_2,c_3} \phi_1^{\mathbf{x}}(c_1, c_2, c_3) \right] \cdot \left[\max_{c_4,c_5} \phi_2^{\mathbf{x}}(c_4, c_5) \right].
\end{aligned}$$

The use of a gray code in each maximal connected component of a CB-decomposable MBC leads to more computational savings in posterior probability computations than without this decomposability. Theorem 10 states those savings and an upper bound of $\frac{F_{GC}}{F_{BF}}$.

Theorem 10 *Given a CB-decomposable MBC with r maximal connected components, where each component i has d_i class variables and m_i feature variables, the independent use of a gray code over the d_i class variables of each component i, can obtain:*

(a) $F_{GC} = \sum_{i=1}^{r}(m_i + d_i + \sum_{j=1}^{d_i} S_j^i H_j^i)$

(b) $\frac{F_{GC}}{F_{BF}} < \frac{1}{|\mathcal{I}|} + \frac{\sum_{i=1}^{r} H_{Max}^i}{m + d}$, *where* $H_{Max}^i = \max_{1 \leq j \leq d_i} H_j^i$, $H_j^i = 1 + h_j^i$, *and* h_j^i *is the number of children of the class variable that changes the jth in the gray code of component i.*

Proof:

(a) is straightforward from Theorem 7. Note that $\sum_{i=1}^{r} m_i = m$, $\sum_{i=1}^{r} d_i = d$.

(b) $F_{GC} \leq \sum_{i=1}^{r}(m_i + d_i + (|\mathcal{I}_i| - 1)H_{Max}^i)$. Then

$$\begin{aligned}
\frac{F_{GC}}{F_{BF}} &\leq \frac{\sum_{i=1}^{r}(m_i + d_i + (|\mathcal{I}_i| - 1)H_{Max}^i)}{(m + d)|\mathcal{I}|} \\
&< \frac{1}{|\mathcal{I}|} + \frac{\sum_{i=1}^{r} |\mathcal{I}_i| H_{Max}^i}{(m + d)|\mathcal{I}|} \\
&< \frac{1}{|\mathcal{I}|} + \frac{\sum_{i=1}^{r} H_{Max}^i}{m + d}
\end{aligned}$$

Example 3 (continued): For the CB-decomposable MBC of Figure 5(a), and considering that all class variables are binary, we have that $m = 6, d = 5, m_1 = 4, d_1 = 3, m_2 = 2, d_2 = 2, |\mathcal{I}| = 32, H_1^1 =$

$3, H^1_2 = 2, H^1_3 = 3, H^2_1 = 2, H^2_2 = 3, H^1_{Max} = 3, H^2_{Max} = 3, S^1_1 = 12, S^1_2 = 4, S^1_3 = 1, S^2_1 = 4$ and $S^2_2 = 1$, and we get:

$$\frac{F_{GC}}{F_{BF}} = \frac{4+3+12\cdot 3+4\cdot 2+1\cdot 3+2+2+4\cdot 2+1\cdot 3}{(6+5)\cdot 32} = \frac{69}{352}$$

and the upper bound is: $\frac{F_{GC}}{F_{BF}} < \frac{1}{32} + \frac{3+3}{6+5} = \frac{203}{352}$, which is a not so good bound than that obtained in Example 1.

4. Bayes decision rule under additive CB-decomposable loss functions

This section extends the previous one, beyond 0-1 loss functions and MPE computations, by providing for other loss functions that conform to CB-decomposable structures.

Definition 11 ***(Additive CB-decomposable loss function)*** *Let $\lambda(\boldsymbol{c}', \boldsymbol{c})$ be a loss function. Given a CB-decomposable MBC B, we say that λ is an* additive CB-decomposable loss function *according to B if*

$$\lambda(\boldsymbol{c}', \boldsymbol{c}) = \sum_{i=1}^{r} \lambda_i(\boldsymbol{c}'^{\downarrow \mathcal{V}_{C_i}}, \boldsymbol{c}^{\downarrow \mathcal{V}_{C_i}}),$$

where λ_i is a non-negative loss function defined on $\mathcal{I}_i$.

Theorem 12 *Let B be a CB-decomposable MBC with r maximal connected components. If λ is an additive CB-decomposable loss function according to B, then*

$$\min_{\boldsymbol{c}' \in \mathcal{I}} R(\boldsymbol{c}'|\boldsymbol{x}) = \sum_{i=1}^{r} \left[\min_{\boldsymbol{c}'^{\downarrow \mathcal{V}_{C_i}} \in \mathcal{I}_i} \sum_{\boldsymbol{c}^{\downarrow \mathcal{V}_{C_i}} \in \mathcal{I}_i} \lambda_i(\boldsymbol{c}'^{\downarrow \mathcal{V}_{C_i}}, \boldsymbol{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \phi_i^{\boldsymbol{x}}(\boldsymbol{c}^{\downarrow \mathcal{V}_{C_i}}) \right] \prod_{j \neq i} \sum_{\boldsymbol{c}^{\downarrow \mathcal{V}_{C_j}} \in \mathcal{I}_j} \phi_j^{\boldsymbol{x}}(\boldsymbol{c}^{\downarrow \mathcal{V}_{C_j}}). \quad (4)$$

Proof.

$$\begin{aligned}
\min_{\mathbf{c}' \in \mathcal{I}} R(\mathbf{c}'|\mathbf{x}) &= \min_{\mathbf{c}' \in \mathcal{I}} \sum_{\mathbf{c} \in \mathcal{I}} \lambda(\mathbf{c}', \mathbf{c}) p(\mathbf{c}|\mathbf{x}) \\
&= \min_{\mathbf{c}' \in \mathcal{I}} \sum_{i=1}^{r} \sum_{\mathbf{c} \in \mathcal{I}} \lambda_i(\mathbf{c}'^{\downarrow \mathcal{V}_{C_i}}, \mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \prod_{j=1}^{r} \phi_j^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_j}}) \\
&= \sum_{i=1}^{r} \min_{\mathbf{c}' \in \mathcal{I}} \sum_{\mathbf{c}^{\downarrow \mathcal{V}_{C_i}} \in \mathcal{I}_i} \lambda_i(\mathbf{c}'^{\downarrow \mathcal{V}_{C_i}}, \mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \phi_i^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \prod_{j \neq i} \sum_{\mathbf{c}^{\downarrow \mathcal{V}_{C_j}} \in \mathcal{I}_j} \phi_j^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_j}}) \\
&= \sum_{i=1}^{r} \min_{\mathbf{c}'^{\downarrow \mathcal{V}_{C_i}} \in \mathcal{I}_i} \sum_{\mathbf{c}^{\downarrow \mathcal{V}_{C_i}} \in \mathcal{I}_i} \lambda_i(\mathbf{c}'^{\downarrow \mathcal{V}_{C_i}}, \mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \phi_i^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_i}}) \cdot \prod_{j \neq i} \sum_{\mathbf{c}^{\downarrow \mathcal{V}_{C_j}} \in \mathcal{I}_j} \phi_j^{\mathbf{x}}(\mathbf{c}^{\downarrow \mathcal{V}_{C_j}})
\end{aligned}$$

The second equality is due to Theorem 9 and because λ is additive CB-decomposable. The third equality takes advantage of the fact that $\lambda \geq 0$ and a grouping of the sums according to the domains of functions $\phi_i^{\mathbf{x}}$ and λ_i. Finally, after the fourth equality, the minimum is computed over the (smaller) spaces given by $\mathcal{I}_i$, where the resulting functions are defined. ■

Corollary 13 *Under the conditions of Theorem 12,*

$$\arg\min_{\boldsymbol{c}'\in\mathcal{I}} R(\boldsymbol{c}'|\boldsymbol{x}) = (\boldsymbol{c}^{*\downarrow\mathcal{V}_{C_1}}, ..., \boldsymbol{c}^{*\downarrow\mathcal{V}_{C_r}}), \tag{5}$$

with $\boldsymbol{c}^{*\downarrow\mathcal{V}_{C_i}} = \arg\min_{\boldsymbol{c}'^{\downarrow\mathcal{V}_{C_i}}\in\mathcal{I}_i} \sum_{\boldsymbol{c}^{\downarrow\mathcal{V}_{C_i}}\in\mathcal{I}_i} \lambda_i(\boldsymbol{c}'^{\downarrow\mathcal{V}_{C_i}}, \boldsymbol{c}^{\downarrow\mathcal{V}_{C_i}}) \cdot \phi_i^{\boldsymbol{x}}(\boldsymbol{c}^{\downarrow\mathcal{V}_{C_i}})$. *This sum, which is to be minimized, is the expected loss over maximal connected component i. Obviously,* $(c_1^*, ..., c_d^*)$ *is readily obtained by assembling the vector in (5) above.*

Proof. The proof is straightforward from Theorem 12, since $\prod_{j\neq i}\sum_{\mathbf{c}^{\downarrow\mathcal{V}_{C_j}}\in\mathcal{I}_j}\phi_j^{\mathbf{x}}(\mathbf{c}^{\downarrow\mathcal{V}_{C_j}})$ in (4) does not depend on i. ■

Example 3 (continued): Let $\mathbf{x}$ be $(0, 0, 0, 0, 0, 0)$. Assume we have the following probabilistic information for the CB-decomposable MBC in Figure 5(a), where all variables (classes and features) are binary:

- For the first maximal connected component:
  - For the class variables: $p(C_1 = 0) = 0.3, p(C_2 = 0) = 0.6, p(C_3 = 0|C_2 = 0) = 0.8, p(C_3 = 0|C_2 = 1) = 0.4$
  - For the features:
    - For X_1, $p(X_1 = 0|C_1 = 0) = 0.2, p(X_1 = 0|C_1 = 1) = 0.3$
    - For X_2, $p(X_2 = 0|C_1 = 0, C_2 = 0, X_1 = 0, X_3 = 0) = 0.2,$
      $p(X_2 = 0|C_1 = 0, C_2 = 1, X_1 = 0, X_3 = 0) = 0.5,$
      $p(X_2 = 0|C_1 = 1, C_2 = 0, X_1 = 0, X_3 = 0) = 0.6,$
      $p(X_2 = 0|C_1 = 1, C_2 = 1, X_1 = 0, X_3 = 0) = 0.3$
    - For X_3, $p(X_3 = 0|C_3 = 0) = 0.5, p(X_3 = 0|C_3 = 1) = 0.7$
    - For X_4, $p(X_4 = 0|C_3 = 0, X_3 = 0, X_5 = 0, X_6 = 0) = 0.7,$
      $p(X_4 = 0|C_3 = 1, X_3 = 0, X_5 = 0, X_6 = 0) = 0.1$
- For the second maximal connected component:
  - For the class variables: $p(C_4 = 0) = 0.6, p(C_5 = 0|C_4 = 0) = 0.3, p(C_5 = 0|C_4 = 1) = 0.1$
  - For the features:
    - For X_5, $p(X_5 = 0|C_4 = 0, C_5 = 0, X_6 = 0) = 0.3,$
      $p(X_5 = 0|C_4 = 0, C_5 = 1, X_6 = 0) = 0.9,$
      $p(X_5 = 0|C_4 = 1, C_5 = 0, X_6 = 0) = 0.8,$
      $p(X_5 = 0|C_4 = 1, C_5 = 1, X_6 = 0) = 0.4$
    - For X_6, $p(X_6 = 0|C_5 = 0) = 0.2, p(X_6 = 0|C_5 = 1) = 0.5$

Let λ be an additive CB-decomposable loss function given by

$$\lambda(c_1', ..., c_5', c_1, ..., c_5) = \lambda_1(c_1', c_2', c_3', c_1, c_2, c_3) + \lambda_2(c_4', c_5', c_4, c_5),$$

where $\lambda_i(\mathbf{c}'^{\downarrow\mathcal{V}_{C_i}}, \mathbf{c}^{\downarrow\mathcal{V}_{C_i}}) = d_H(\mathbf{c}'^{\downarrow\mathcal{V}_{C_i}}, \mathbf{c}^{\downarrow\mathcal{V}_{C_i}})$, with $i = 1, 2$, and d_H denotes the Hamming distance, i.e. the number of coordinates where $\mathbf{c}'^{\downarrow\mathcal{V}_{C_i}}$ and $\mathbf{c}^{\downarrow\mathcal{V}_{C_i}}$ are different. Note that in our multi-dimensional classification problem, d_H counts the total number of errors made by the classifier in the class variables. Thus, λ_1 is

(c_1, c_2, c_3)	$\phi_1^{\mathbf{x}}(c_1, c_2, c_3)$	(c'_1, c'_2, c'_3)	$\sum_{c_1,c_2,c_3} \lambda_1 \phi_1^{\mathbf{x}},$
(0, 0, 0)	0.0020160	(0, 0, 0)	0.0363048
(0, 0, 1)	0.0001008	(0, 0, 1)	0.0831432
(0, 1, 0)	0.0016800	(0, 1, 0)	0.0538776
(0, 1, 1)	0.0005040	(0, 1, 1)	0.0795480
(1, 0, 0)	0.0211680	(1, 0, 0)	0.0275016
(1, 0, 1)	0.0010584	(1, 0, 1)	0.0394632
(1, 1, 0)	0.0035280	(1, 1, 0)	0.0313656
(1, 1, 1)	0.0010584	(1, 1, 1)	0.0570360

Table 1: Computing the minimum expected loss in the first maximal connected component

λ_1	(0,0,0)	(0,0,1)	(0,1,0)	(0,1,1)	(1,0,0)	(1,0,1)	(1,1,0)	(1,1,1)
(0,0,0)	0	1	1	2	1	2	2	3
(0,0,1)	1	0	2	1	3	1	3	2
(0,1,0)	1	2	0	1	2	3	1	2
(0,1,1)	2	1	1	0	3	2	2	1
(1,0,0)	1	3	2	3	0	1	1	2
(1,0,1)	2	1	3	2	1	0	2	1
(1,1,0)	2	3	1	2	1	2	0	1
(1,1,1)	3	2	2	1	2	1	1	0

and λ_2 is

λ_2	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	0	1	1	2
(0,1)	1	0	2	1
(1,0)	1	2	0	1
(1,1)	2	1	1	0

We have that

$$\begin{aligned}\phi_1^{\mathbf{x}}(c_1, c_2, c_3) &= p(c_1)p(c_2)p(c_3|c_2) \\ &\cdot\ p(X_1 = 0|c_1)p(X_2 = 0|c_1, c_2, X_1 = 0, X_3 = 0)p(X_3 = 0|c_3) \\ &\cdot\ p(X_4 = 0|c_3, X_3 = 0, X_5 = 0, X_6 = 0)\end{aligned}$$

Table 1 lists the whole set of $\phi_1^{\mathbf{x}}$ values on the left-hand side. The rest of the table develops the computations required for $\mathbf{c}^{*\downarrow\mathcal{V}_{C_i}}$ as indicated in Corollary 13. Therefore, $\mathbf{c}^{*\downarrow\{C_1,C_2,C_3\}} = (1, 0, 0)$.

Furthermore, we have that

$$\phi_2^{\mathbf{x}}(c_4, c_5) = p(c_4)p(c_5|c_4)p(X_5 = 0|c_4, c_5, X_6 = 0)p(X_6 = 0|c_5),$$

where the associated results are shown in Table 2. Therefore, $\mathbf{c}^{*\downarrow\{C_4,C_5\}} = (0, 1)$.

Finally, our CB-decomposable MBC assigns the class vector $\mathbf{c}^* = (1, 0, 0, 0, 1)$ to the feature vector $\mathbf{x} = (0, 0, 0, 0, 0, 0)$.

(c_4, c_5)	$\phi_2^{\mathbf{x}}(c_4, c_5)$	(c_4', c_5')	$\sum_{c_4,c_5} \lambda_2 \phi_2^{\mathbf{x}}$
(0, 0)	0.0108	(0, 0)	0.3394
(0, 1)	0.1890	(0, 1)	0.0956
(1, 0)	0.0064	(1, 0)	0.4608
(1, 1)	0.0720	(1, 1)	0.2170

Table 2: Computing the minimum expected loss in the second maximal connected component

Corollary 14 *Under the same assumptions as in Theorem 12 with $r = d$ maximal connected components, then*

$$\min_{\boldsymbol{c}' \in \mathcal{I}} R(\boldsymbol{c}'|\boldsymbol{x}) = \sum_{i=1}^{d} \left[\min_{c_i'} \sum_{c_i} \lambda_i(c_i', c_i) \cdot \phi_i^{\boldsymbol{x}}(c_i) \right] \prod_{j \neq i} \sum_{c_j} \phi_j^{\boldsymbol{x}}(c_j),$$

where

$$\phi_i^{\boldsymbol{x}}(c_i) = p(c_i) \prod_{X \in Ch(C_i)} p(x|c_i, \boldsymbol{pa}_{\mathcal{V}_X}(x)).$$

Proof. The proof is straightforward from Theorem 12. Under this decomposability, class variables are not longer conditioned to other class variables. ■

Note that the simplest CB-decomposability applies in this case.

5. Performance Evaluation Metrics for Multi-Dimensional Classifiers

We propose the following performance measures that extend metrics existing in the single-class domain. Cases $i \in \{1, ..., N\}$ are assumed to belong to the test data set.

1. Mean accuracy over the d class variables:

   $$\overline{Acc_d} = \frac{1}{d} \sum_{j=1}^{d} Acc_j = \frac{1}{d} \sum_{j=1}^{d} \frac{1}{N} \sum_{i=1}^{N} \delta(c_{ij}', c_{ij}), \tag{6}$$

   where $\delta(c_{ij}', c_{ij}) = 1$ if $c_{ij}' = c_{ij}$, and 0 otherwise. Note that c_{ij}' denotes the C_j class value outputted by the model for case i and c_{ij} is its true value.

   A similar concept may be extended to CB-decomposable MBCs by means of the mean accuracy over the r maximal connected components: $\overline{Acc_r} = \frac{1}{r} \sum_{j=1}^{r} \frac{1}{N} \sum_{i=1}^{N} \delta(\mathbf{c}_i'^{\downarrow \mathcal{V}_{C_j}}, \mathbf{c}_i^{\downarrow \mathcal{V}_{C_j}})$.

2. Global accuracy over the d-dimensional class variable:

   $$Acc = \frac{1}{N} \sum_{i=1}^{N} \delta(\mathbf{c}_i', \mathbf{c}_i) \tag{7}$$

   where $\delta(\mathbf{c}_i', \mathbf{c}_i) = 1$ if $\mathbf{c}_i' = \mathbf{c}_i$, and 0 otherwise. Therefore, we call for a total coincidence in all the components of the vector of predicted classes and the vector of real classes.

It holds that $Acc \leq \overline{Acc_r} \leq \overline{Acc_d}$. This is obvious since it is less demanding to count errors in a component-wise fashion than as a vector of components that, as a whole, has to correctly predict all its coordinates.

If all the class variables are binary, then we also have:

3. Micro $F1$:

$$\text{Micro } F1 = 2\frac{\overline{Pre_d} \times \overline{Rec_d}}{\overline{Pre_d} + \overline{Rec_d}}, \tag{8}$$

where mean precision and mean recall over the d class variables are defined, respectively, as

$$\overline{Pre_d} = \frac{1}{d}\sum_{j=1}^{d} Pre^j, \qquad \overline{Rec_d} = \frac{1}{d}\sum_{j=1}^{d} Rec^j.$$

Pre^j and Rec^j are the precision and recall, respectively, obtained from the single-class confusion matrix of C_j, i.e. $Pre^j = \frac{TP^j}{TP^j+FP^j}, Rec^j = \frac{TP^j}{TP^j+FN^j}$, where TP^j, FN^j, TN^j, FP^j are the counts for true positives, false negatives, true negatives and false positives, respectively.

4. Macro $F1$:

$$\text{Macro } F1 = 2\frac{Pre^g \times Rec^g}{Pre^g + Rec^g}, \tag{9}$$

where

$$Pre^g = \frac{\sum_{j=1}^{d} TP^j}{\sum_{j=1}^{d}(TP^j + FP^j)}, \qquad Rec^g = \frac{\sum_{j=1}^{d} TP^j}{\sum_{j=1}^{d}(TP^j + FN^j)},$$

which could be seen as global precision and global recall.

6. Learning MBCs from Data

In this section we introduce algorithms to learn MBCs from data. Let $\mathcal{D}$ be a database of N observations containing a value assignment for each variable $X_1, ..., X_m, C_1, ..., C_d$, i.e. $\mathcal{D} = \{(\mathbf{x}^{(1)}, \mathbf{c}^{(1)}), ..., (\mathbf{x}^{(N)}, \mathbf{c}^{(N)})\}$. The learning problem is to find an MBC that best fits the available data. We will use a score + search approach (Cooper and Herskovits, 1992) to find the MBC structure. MBC parameters can be estimated as in standard Bayesian networks. The *score* measuring the goodness of an MBC given $\mathcal{D}$ can be independent of or dependent on the classifier performance measure (a filter score or a wrapper score respectively) (Kohavi and John, 1997). Although any kind of strategy could be employed to *search* the MBC space, the algorithms proposed below follow a greedy search for computational reasons. The algorithms will, however, be flexible due to the possibility of incorporating filter, wrapper or hybrid approaches and because, as opposed to other proposals found in the literature, any kind of structure is allowed for the class and feature subgraphs.

6.1 PURE FILTER Algorithm

Given a fixed ordering of all the variables $O = (O_C, O_X) = (C_{\pi(1)}, ..., C_{\pi(d)}, X_{\pi'(1)}, ..., X_{\pi'(m)})$, where π and π' are permutations over the variables in $\mathcal{V}_C$ and $\mathcal{V}_X$, respectively, this algorithm first learns the class subgraph, $\mathcal{G}_C$, with a filter score that takes into account ordering O_C and then learns the feature subgraph, $\mathcal{G}_X$, using O_X in the same way, once the bridge subgraph, $\mathcal{G}_{CX}$, is fixed.

The learning problem can be solved as two separate problems: (1) the search for the best structure of $\mathcal{G}_C$, taking into account only the $\mathcal{D}_C = \{\mathbf{c}^{(1)}, ..., \mathbf{c}^{(N)}\}$ values, which is solved once; and (2) the search for the best structure of $\mathcal{G}_X$, constrained by fixed parents in $\mathcal{V}_C$ given in a candidate bridge subgraph $\mathcal{G}_{CX}$, which is then updated via a best-first search in $\mathcal{G}_{CX}$. By choosing a decomposable score, both searches, in $\mathcal{G}_X$ and $\mathcal{G}_{CX}$, may reduce their computational burden considerably since only local computations are carried out.

Considering the MBC structure, the global score $s(\mathcal{D}|\mathcal{G})$ to be maximized, is the sum of the score over the class variables, $s(\mathcal{D}_C|\mathcal{G}_C)$, and the score over the feature variables given a fixed structure $\mathcal{G}_{CX}$ from classes to features, $s_{\mathcal{G}_{CX}}(\mathcal{D}|\mathcal{G}_X)$.

The algorithm is shown below.

```
PURE FILTER algorithm
Input: An ordering $O = (O_C, O_X)$
   $i = 0$
   1. [Learn class subgraph]
      Learn $\mathcal{G}_C^*$ with a filter score, $s(\mathcal{D}_C|\mathcal{G}_C)$ based on $O_C$.
   2. [Learn feature subgraph]
      Set ${\mathcal{G}_{CX}}^{(0)} = \varnothing$. Learn ${\mathcal{G}_X}^{(0)}$ with a filter score $s_{{\mathcal{G}_{CX}}^{(0)}}(\mathcal{D}|{\mathcal{G}_X}^{(0)})$, based on $O_X$.
   3. [Propose a candidate bridge subgraph]
      Add one arc to ${\mathcal{G}_{CX}}^{(i)}$ to get a candidate ${\mathcal{G}_{CX}}^{(i+1)}$.
   4. [Obtain a candidate feature subgraph]
      Given the candidate ${\mathcal{G}_{CX}}^{(i+1)}$, learn ${\mathcal{G}_X}^{(i+1)}$ with the filter score $s_{{\mathcal{G}_{CX}}^{(i+1)}}(\mathcal{D}|{\mathcal{G}_X}^{(i+1)})$,
      constrained by ${\mathcal{G}_{CX}}^{(i+1)}$, based on $O_X$.
   5. [Decide on the bridge and feature subgraph candidates]
      If $s_{{\mathcal{G}_{CX}}^{(i+1)}}(\mathcal{D}|{\mathcal{G}_X}^{(i+1)}) > s_{{\mathcal{G}_{CX}}^{(i)}}(\mathcal{D}|{\mathcal{G}_X}^{(i)})$, update the bridge subgraph, $i = i + 1$ and go to 3
         Else, while there are unvisited candidates, go to 3.
         Otherwise, $\mathcal{G}_X^* = {\mathcal{G}_X}^{(i+1)}$ and $\mathcal{G}_{CX}^* = {\mathcal{G}_{CX}}^{(i+1)}$, and stop.
Output: $\mathcal{G}^* = \mathcal{G}_C^* \cup \mathcal{G}_X^* \cup \mathcal{G}_{CX}^*$.
```

Note that the MBC structure and the ancestral fixed order $O = (O_C, O_X)$ allow us to use data from the class variables, to search for the best class subgraph, $\mathcal{G}_C^*$, independently of the other variables (step 1). On the other hand, we organize the search of the rest of the graph by first fixing a bridge subgraph (steps 2 and 5) and then searching for the best feature subgraph conditioned to the class parents given in the bridge subgraph (step 4). An example would be to apply a K2 algorithm over the features using O_X but with the bridge subgraph imposing some class variables as parents. Then, we move to a new bridge subgraph which is equal to the previous one except for one added arc (step 3). This greedy strategy is handy from a computational point of view, since if we use a decomposable score, the new and old scores only differ by the term involving the new arc. That is, when arc $(C_l, X_j) \in \mathcal{A}_{CX}$ is added to the bridge subgraph ${\mathcal{G}_{CX}}^{(i)}$ to have a candidate bridge subgraph

${\mathcal{G}_{CX}}^{(i+1)}$, then the difference required in step 5, $s_{{\mathcal{G}_{CX}}^{(i+1)}}(\mathcal{D}|{\mathcal{G}_X}^{(i+1)}) - s_{{\mathcal{G}_{CX}}^{(i)}}(\mathcal{D}|{\mathcal{G}_X}^{(i)})$, consists of the score only evaluating X_j and its new parents.

Note that the greedy strategy is forward, starting from the empty graph (step 2). Each time a better structure (bridge + features) is found (a better score), we update the current structure with this new one –a best-first strategy– and start the forward scheme from here. The process stops when any addition to the current bridge subgraph fails to provide a feature subgraph that improves the score.

6.2 pure wrapper Algorithm

This algorithm greedily searches for one arc, to be added or removed, in any position but respecting the MBC structure, such that the global accuracy *Acc*, as defined in Section 5, is improved. Any general DAG structure is allowed for the class and feature subgraphs. The algorithm stops if no arcs can be added or deleted to the current structure to improve the global accuracy.

The algorithm is shown below.

```
pure wrapper algorithm
$i = 0$
1. $\mathcal{G}^{(i)} = \emptyset$. $Acc = Acc^{(i)}$.
2. Whenever there are arcs that can be added to $\mathcal{G}^{(i)}$ (and not previously discarded):
Add/delete one arc to ${\mathcal{G}_C}^{(i)}$, ${\mathcal{G}_{CX}}^{(i)}$ or ${\mathcal{G}_X}^{(i)}$ and obtain the new $\mathcal{G}^{(i+1)}$ and $Acc^{(i+1)}$.
3. If $Acc^{(i+1)} > Acc^{(i)}$, $Acc = Acc^{(i+1)}$, $i = i + 1$, and go to 2.
      Else discard the arc and go to 2.
4. Stop and return $\mathcal{G}^{(i)}$ and $Acc$.
```

This algorithm is controlled by the *Acc* measure. Any other performance measure defined in Section 5 could be used. However, computing *Acc* involves the computation of the MPE for the class variables given the features, and this has the advantage of being alleviated if there are CB-decomposable MBCs (Theorem 9), which is likely to be the case as the algorithm progresses, specially in the early stages. Also, gray codes will reduce the computations (Theorems 10 and 7).

Special case of an additive CB-decomposable loss function. Let $\lambda(\mathbf{c}', \mathbf{c})$ be an additive CB-decomposable loss function according to a CB-decomposable MBC B with r maximal connected components. Then the pure wrapper algorithm can be applied with the following modifications. The global accuracy *Acc* counts the number of correctly classified cases based on the real and predicted class vectors. However, since the loss function is no any longer 0-1, the predicted class vector for each $\mathbf{x}$ is obtained by minimizing its expected loss $R(\mathbf{c}|\mathbf{x})$. When λ is additive CB-decomposable, Theorems 10 and 12 and Corollary 13 provide computational savings via gray codes and MBC decomposability, which would be beneficial in step 2 of the algorithm. Moreover, when trying to move to a new structure at step 2, if λ is additive CB-decomposable, the added arc should guarantee that a CB-decomposable MBC with r maximal connected components according to λ is yielded. This means that the class subgraph is constrained by only allowing arcs among class variables of the same group $\mathcal{V}_{Ci}$, $i = 1, ..., r$ defined by λ. It also means that the non-shared children property for the r components should hold. Note that our forward strategy starting from the empty structure will produce structures with $h > r$ maximal connected components in the early iterations. These structures will be valid as long as they do not contradict the groups of class variables given by λ.

6.3 hybrid Algorithm

This algorithm is equal to the pure filter algorithm but the decision on the candidate structures at step 5 is made based on the global accuracy Acc (or any other performance measure), rather than on a general score s.

6.4 Cardinality of MBC Structure Space

The above learning algorithms move within the MBC structure space. Thus, knowledge of the cardinality of this space can help us to infer the complexity of the learning problem. We will point out two cases. The first one is the general MBC, whereas the second one places two constraints on the MBC bridge subgraph sometimes found in the literature (van der Gaag and de Waal, 2006; de Waal and van der Gaag, 2007).

Theorem 15 *The number of all possible MBC structures with d class variables and m feature variables, $MBC(d,m)$, is*

$$MBC(d,m) = S(d) \cdot 2^{dm} \cdot S(m),$$

where $S(n) = \sum_{i=1}^{n}(-1)^{i+1}\binom{n}{i}2^{i(n-i)}S(n-i)$ is Robinson's formula (Robinson, 1973) that counts the number of possible DAG structures of n nodes, which is initialized as $S(0) = S(1) = 1$.

Proof. $S(d)$ and $S(m)$ count the possible DAG structures for the class subgraph and feature subgraph, respectively. 2^{dm} is the number of possible bridge subgraphs. ■

We now consider MBCs satisfying the following conditions on their bridge subgraph: (a) for each $X_i \in \mathcal{V}_X$, there is a $C_j \in \mathcal{V}_C$ with $(C_j, X_i) \in \mathcal{A}_{CX}$ and (b) for each $C_j \in \mathcal{V}_C$, there is an $X_i \in \mathcal{V}_X$ with $(C_j, X_i) \in \mathcal{A}_{CX}$. These conditions were used in van der Gaag and de Waal (2006) and in de Waal and van der Gaag (2007) for learning `tree-tree` and `polytree-polytree` MBCs, respectively. The number of possible bridge subgraphs is given by the following theorem.

Theorem 16 *The number of all possible bridge subgraphs, $BRS(d,m)$, $m \geq d$, for MBCs satisfying the two previous conditions (a) and (b) is given by the recursive formula*

$$BRS(d,m) = 2^{dm} - \sum_{k=0}^{m-1}\binom{dm}{k} - \sum_{k=m}^{dm}\sum_{\substack{x\leq d, y\leq m\\ k\leq xy\leq dm-d}}\binom{d}{x}\binom{m}{y}BRS(x,y,k),$$

where $BRS(x,y,k)$ denotes the number of bridge subgraphs with k arcs in an MBC with x class variables and y feature variables which is initialized as $BRS(1,1,1) = BRS(1,2,2) = BRS(2,1,2) = 1$.

Proof. It holds that

$$\begin{aligned} BRS(d,m) &= \sum_{k=\max\{d,m\}=m}^{dm} BRS(d,m,k) \\ &= \sum_{k=m}^{dm}\left[\binom{dm}{k} - \sum_{\substack{x\leq d, y\leq m\\ k\leq xy\leq dm-d}}\binom{d}{x}\binom{m}{y}BRS(x,y,k)\right] \\ &= 2^{dm} - \sum_{k=0}^{m-1}\binom{dm}{k} - \sum_{k=m}^{dm}\sum_{\substack{x\leq d, y\leq m\\ k\leq xy\leq dm-d}}\binom{d}{x}\binom{m}{y}BRS(x,y,k) \end{aligned}$$

The first equality is true since resulting MBCs must satisfy the two conditions on the bridge subgraphs, requiring at least $k = \max\{d, m\} = m$ arcs. In the second equality, $BRS(d, m, k)$ is computed by subtracting the bridge subgraphs not satisfying the two required conditions from the number of possible bridge subgraphs with k arcs, which is $\binom{dm}{k}$. These "invalid" bridge subgraphs include arcs from x class variables to y feature variables, such that $x \leq d, y \leq m$ and $k \leq xy \leq dm-d$. xy must be at least k to have k arcs in the bridge subgraph. Also, xy must be lower than $dm - d + 1$, which is the maximum number of arcs for a valid bridge subgraph. Finally, the third equality is straightforward from the expansion of $(1 + 1)^{dm}$. ■

Theorem 17 *The number of all possible MBC structures with d class variables and m feature variables, $m \geq d$, satisfying conditions (a) and (b), $MBC^{ab}(d, m)$ is*

$$MBC^{ab}(d, m) = S(d) \cdot BRS(d, m) \cdot S(m).$$

Proof. The proof is straightforward from Theorem 15 and Theorem 16. ■

Corollary 18 *The number of all possible MBC structures with d class variables and m feature variables satisfies*

$$O(MBC(d, m)) = O(MBC^{ab}(d, m)) = 2^{dm}(\max\{d, m\})^{2^{O(\max\{d,m\})}}.$$

Proof. The complexity of Robinson's formula (Robinson, 1973) was shown to be super exponential, i.e. $O(S(n)) = n^{2^{O(n)}}$. Also, $O(BRS(d, m)) = 2^{dm}$. Therefore,

$$\begin{aligned} O(MBC(d, m)) &= O(MBC^{ab}(d, m)) \\ &= d^{2^{O(d)}} \cdot 2^{dm} \cdot m^{2^{O(m)}} \\ &\leq (\max\{d, m\})^{2 \cdot 2^{O(\max\{d,m\})}} \cdot 2^{dm} \\ &= 2^{dm}(\max\{d, m\})^{2^{O(\max\{d,m\})}} \end{aligned}$$

■

7. Experimental Results on MPE

An upper bound for the number of factors saved when computing the posterior probabilities of the MPE with gray codes was given in Section 3. This obviously has an effect on the required time. Here we compare the efficiency of the gray codes against a brute force approach as exact algorithms for MPE computation.

The experiment consists of randomly generating 12 different MBCs with a number of binary class variables ranging from $d = 3$ to $d = 14$ and with $m = 10$ feature variables. We then compute ten MPE problems as in (2), given ten random evidences $\mathbf{x}^{(1)}, ..., \mathbf{x}^{(10)}$. For the gray codes, computations are based on Corollary 4.

Figure 6 shows error bars for computation times when using both exact approaches. They are obtained from the average times over the ten MPE problems minus/plus the standard deviation.

Note that the gray code approach is faster than brute force, and this effect is more significant as the number of class variables, d, increases. This is consistent with the bounds computed in Section 3, since $\mathcal{I}$ and d appear in the denominator of Theorem 7.

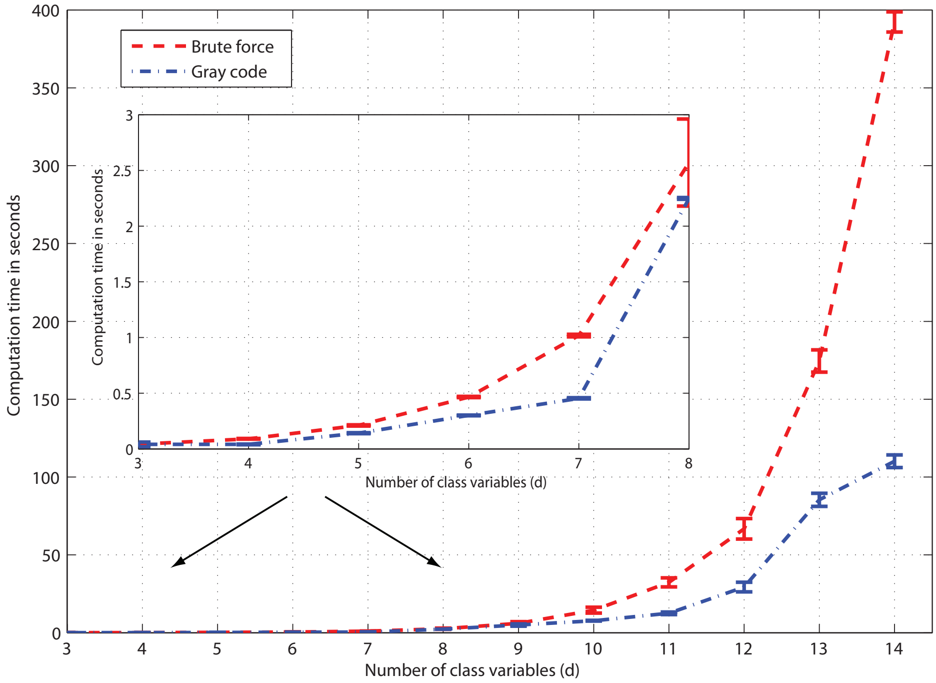


Figure 6: Error bars for running times when MPE is computed with gray code (blue) and brute force (red) approaches

When the number of class variables increases a lot, we must resort to approximate algorithms, for example taboo search, hill climbing and sequential initialization as recommended in Park and Darwiche (2004).

8. Experimental Results on Learning MBCs

8.1 Data Sets

For the purpose of our study, we use three benchmark data sets[2]. `Emotions` data set (Trohidis et al., 2008) includes 593 sound clips from a 30-seconds sequence after the initial 30 seconds of a song. The 72 features extracted fall into two categories: 8 rhythmic features and 64 timbre features. Songs are categorized by six class variables: amazed-surprised, happy-pleased, relaxing-calm, quiet-still, sad-lonely, and angry-aggressive.

The `Scene` data set (Boutell et al., 2004) has 2407 pictures, and their semantic scenes have to be classified into six class binary variables: beach, sunset, foliage, field, mountain, and urban. The 294 features correspond to spatial color moments in the LUV space.

The `Yeast` data set (Elisseeff and Weston, 2002) is about predicting the 14 functional classes of 2417 genes in the Saccharomyces Cerevisae Yeast. Each gene is described by the concatenation of microarray expression data and a phylogenetic profile given by 103 features.

All class variables are binary. The details of the three data sets are summarized in Table 3.

[2] Databases are available at: `http://mlkd.csd.auth.gr/multilabel.html`

Data set	Domain	Size	m	d
`Emotions`	Music	593	72	6
`Scene`	Vision	2407	294	6
`Yeast`	Biology	2417	103	14

Table 3: Basic information of the three data sets

However, feature variables are numeric. Since MBCs are defined for discrete variables, it is necessary to discretize all the continuous features. We use a static, global, supervised and top-down discretization algorithm called class-attribute contingency coefficient (Cheng-Jung et al., 2008). The `Emotions` and `Scene` data sets contain some missing records. Only their no-missing parts are used in the computation of (conditional) probabilities.

8.2 Experimental Setup

We use eight different algorithms to learn MBCs. First, we apply five algorithms explicitly designed for MBCs: TREE-TREE (van der Gaag and de Waal, 2006), POLYTREE-POLYTREE (de Waal and van der Gaag, 2007) and PURE FILTER, PURE WRAPPER and HYBRID described in Section 6. Second, we use two greedy search algorithms that learn a general Bayesian network, one guided by the K2 metric (Cooper and Herskovits, 1992) (filter approach), and the other guided by a performance evaluation metric, as defined in Section 5 (wrapper approach). The first one will be denoted K2 BN, while the second one will be WRAPPER BN. Third, we consider a multi-label lazy learning approach named ML-KNN (Zhang and Zhou, 2007), see Section 9.1, derived from the traditional K-nearest neighbor algorithm. In this case, we set K to 3 in the `Emotions` and `Scene` data sets, and 5 in the `Yeast` data set. As explained in Section 9.4, since it is unfeasible to compute the mutual information of two features given *all* the class variables, as required in de Waal and van der Gaag (2007), we decided to implement the POLYTREE-POLYTREE learning algorithm using the (marginal) mutual information of pairs of features. The heuristic searches always terminate after 200 unsuccessful trials looking for better structures.

The probabilities attached to each node in the learnt network are calculated by maximum likelihood estimation, corrected with Laplace smoothing. As regards MPE computations required in the performance evaluation metrics, we use gray codes for the `Emotions` and `Scene` data sets and due to its big size, an approximate algorithm called sequential initialization (Park and Darwiche, 2004) for the `Yeast` data set. The estimation method for performance evaluation metrics is 10-fold cross validation (Stone, 1974).

8.3 Results

Table 4 shows the results of the eight algorithms over the three data sets. Mean values and standard deviations are listed for each metric. The number in brackets is the rank of the algorithm in decreasing order of performance (i.e. 1=best performance, 8=worst performance).

The average ranking of the eight algorithms is presented in Table 5. The algorithms are ordered as follows: PURE WRAPPER $\succ$ TREE-TREE $\succ$ WRAPPER BN $\succ$ POLYTREE-POLYTREE $\succ$ K2 BN $\succ$ PURE FILTER $\succ$ HYBRID $\succ$ ML-KNN.

There are several non-parametric statistical tests and procedures to compare the performance of classifiers over multiple data sets. Following García and Herrera (2008), we recommend Ne-

	Mean Accuracy	Global Accuracy	Micro F1	Macro F1
`Emotions`				
tree-tree	0.8300 ± 0.0151 (2)	0.3844 ± 0.0398 (1)	0.7921 ± 0.0273 (3)	0.8072 ± 0.0225 (2)
polytree-polytree	0.8209 ± 0.0243 (4)	0.3776 ± 0.0622 (2)	0.7829 ± 0.0264 (4)	0.7915 ± 0.0329 (4)
pure filter	0.7548 ± 0.0280 (7)	0.2866 ± 0.0495 (6)	0.7106 ± 0.0363 (7)	0.7243 ± 0.0370 (7)
pure wrapper	0.8333 ± 0.0123 (1)	0.3708 ± 0.0435 (3)	0.9145 ± 0.1107 (2)	0.8077 ± 0.0189 (1)
hybrid	0.8210 ± 0.0170 (3)	0.3557 ± 0.0435 (4)	0.7580 ± 0.0342 (5)	0.7898 ± 0.0446 (5)
k2 bn	0.7751 ± 0.0261 (6)	0.2812 ± 0.0799 (7)	0.7315 ± 0.0351 (6)	0.7429 ± 0.0363 (6)
wrapper bn	0.7985 ± 0.0200 (5)	0.3033 ± 0.0752 (5)	1.0000 ± 0.0000 (1)	0.7932 ± 0.0284 (3)
ml-knn	0.6133 ± 0.0169 (8)	0.0254 ± 0.0120 (8)	0.3428 ± 0.0368 (8)	0.3385 ± 0.0455 (8)
`Scene`				
tree-tree	0.7324 ± 0.0359 (7)	0.1857 ± 0.0977 (6)	0.3705 ± 0.1110 (5)	0.3465 ± 0.1256 (6)
polytree-polytree	0.7602 ± 0.0663 (6)	0.2643 ± 0.1915 (4)	0.3942 ± 0.1362 (4)	0.4181 ± 0.2105 (5)
pure filter	0.7726 ± 0.0700 (4)	0.3067 ± 0.1991 (1)	0.3494 ± 0.1263 (7)	0.4560 ± 0.2161 (3)
pure wrapper	0.7765 ± 0.0580 (2)	0.2688 ± 0.1642 (3)	1.0000 ± 0.0000 (1)	0.4893 ± 0.2436 (1)
hybrid	0.7229 ± 0.0442 (8)	0.1570 ± 0.1018 (7)	0.2593 ± 0.1491 (8)	0.2979 ± 0.1500 (7)
k2 bn	0.7689 ± 0.0692 (5)	0.2883 ± 0.1995 (2)	0.3571 ± 0.1272 (6)	0.4477 ± 0.2235 (4)
wrapper bn	0.7739 ± 0.0492 (3)	0.2277 ± 0.1372 (5)	0.9446 ± 0.1751 (2)	0.4612 ± 0.2029 (2)
ml-knn	0.8196 ± 0.0092 (1)	0.0311 ± 0.0147 (8)	0.6055 ± 0.5076 (3)	0.0567 ± 0.0233 (8)
`Yeast`				
tree-tree	0.7728 ± 0.0071 (3)	0.1953 ± 0.0207 (1)	0.5009 ± 0.0203 (2)	0.6910 ± 0.0131 (2)
polytree-polytree	0.7336 ± 0.0182 (7)	0.1431 ± 0.0257 (2)	0.4993 ± 0.0325 (3)	0.6517 ± 0.0208 (7)
pure filter	0.7480 ± 0.0119 (5)	0.0989 ± 0.0342 (6)	0.4335 ± 0.0092 (6)	0.6679 ± 0.0169 (4)
pure wrapper	0.7845 ± 0.0131 (1)	0.1410 ± 0.0989 (3)	0.5287 ± 0.0221 (1)	0.6957 ± 0.0099 (1)
hybrid	0.7397 ± 0.0114 (6)	0.1200 ± 0.0268 (5)	0.4302 ± 0.0154 (7)	0.6580 ± 0.0162 (5)
k2 bn	0.7686 ± 0.0112 (4)	0.1299 ± 0.0204 (4)	0.4498 ± 0.0160 (4)	0.6830 ± 0.0234 (3)
wrapper bn	0.7745 ± 0.0049 (2)	0.0550 ± 0.0212 (7)	0.4379 ± 0.0206 (5)	0.6575 ± 0.0228 (6)
ml-knn	0.6364 ± 0.0196 (8)	0.0062 ± 0.0029 (8)	0.3077 ± 0.0273 (8)	0.3218 ± 0.0460 (8)

Table 4: Estimated performance metrics (mean ± std deviation) and rank (in brackets) of the eight learning algorithms over the `Emotions`, `Scene` and `Yeast` data sets using 10-fold cross validation

menyi's test, and Holm's, Shaffer's static and Bergmann-Hommel's procedures to conduct all pairwise comparisons in a multiple comparison analysis. The authors detail how to obtain adjusted and comparable p-values in such multiple comparison procedures.

The adjusted p-values are compared against a significance level of $\alpha = 0.05$ to reject or accept the null hypothesis stating that a pair of algorithms perform equally. By observing the significance of the tests in Holm's, Shaffer's static and Bergmann-Hommel's procedures, which are the most powerful, the conclusions are: (1) pure wrapper turns out to be significantly better than k2 bn, pure filter, hybrid and ml-knn; (2) tree-tree is significantly better than ml-knn; and (3) wrapper bn is better than ml-knn. Nemenyi's procedure provides the same results given in (1), (2) and (3), except for the pair pure wrapper and k2 bn, whose difference in performance is not statistically significant. Note that for these data sets, all concerning multi-label classification, the state-of-the-art algorithm ml-knn has been beaten by three algorithms not specifically designed for a multi-label setting.

	Mean Accuracy	Global Accuracy	Micro F1	Macro F1	Ave ranking
TREE-TREE	4.00	2.66	3.33	3.33	3.33
POLYTREE-POLYTREE	5.66	2.66	3.66	5.33	4.33
PURE FILTER	5.33	4.33	6.66	4.66	5.25
PURE WRAPPER	1.33	3.00	1.33	1.00	1.66
HYBRID	5.66	5.33	6.66	5.66	5.83
K2 BN	5.00	4.33	5.33	4.33	4.75
WRAPPER BN	3.33	5.66	2.66	3.66	3.83
ML-KNN	5.66	8.00	6.33	8.00	7.00

Table 5: Average rankings of the eight algorithms over four metrics and three data sets

In short, one of the new algorithms proposed here, PURE WRAPPER, TREE-TREE (van der Gaag and de Waal, 2006), and the general WRAPPER BN turn out to be the best algorithms, where PURE WRAPPER is the most outstanding of the three.

We present some examples of the networks learnt with the best three algorithms. Figure 7 shows three networks learnt from the `Emotions` data set: with PURE WRAPPER, TREE-TREE and WRAPPER BN algorithms. Red nodes represent class variables, yellow nodes are feature variables, red arrows represent arcs in the class subgraph, blue arrows represent arcs in the feature subgraph and green arrows represent arcs in the bridge subgraph. Note that arrows from features to class variables are allowed in WRAPPER BN. They are shown in gray.

Note that, in these examples, PURE WRAPPER and WRAPPER BN yield sparser networks than TREE-TREE. This behavior holds in general for the other data sets and performance evaluation metrics.

Finally, computation times of the learning process with the eight algorithms are shown in Figure 8. Figure 8 illustrates the computation time of each algorithm when using each performance metric for each data set. TREE-TREE takes the longest, whereas ML-KNN is the fastest. Generally speaking, the algorithms using a filter approach, such as POLYTREE-POLYTREE, K2 BN and PURE FILTER, require less computation, whereas those using a wrapper approach take more computation.

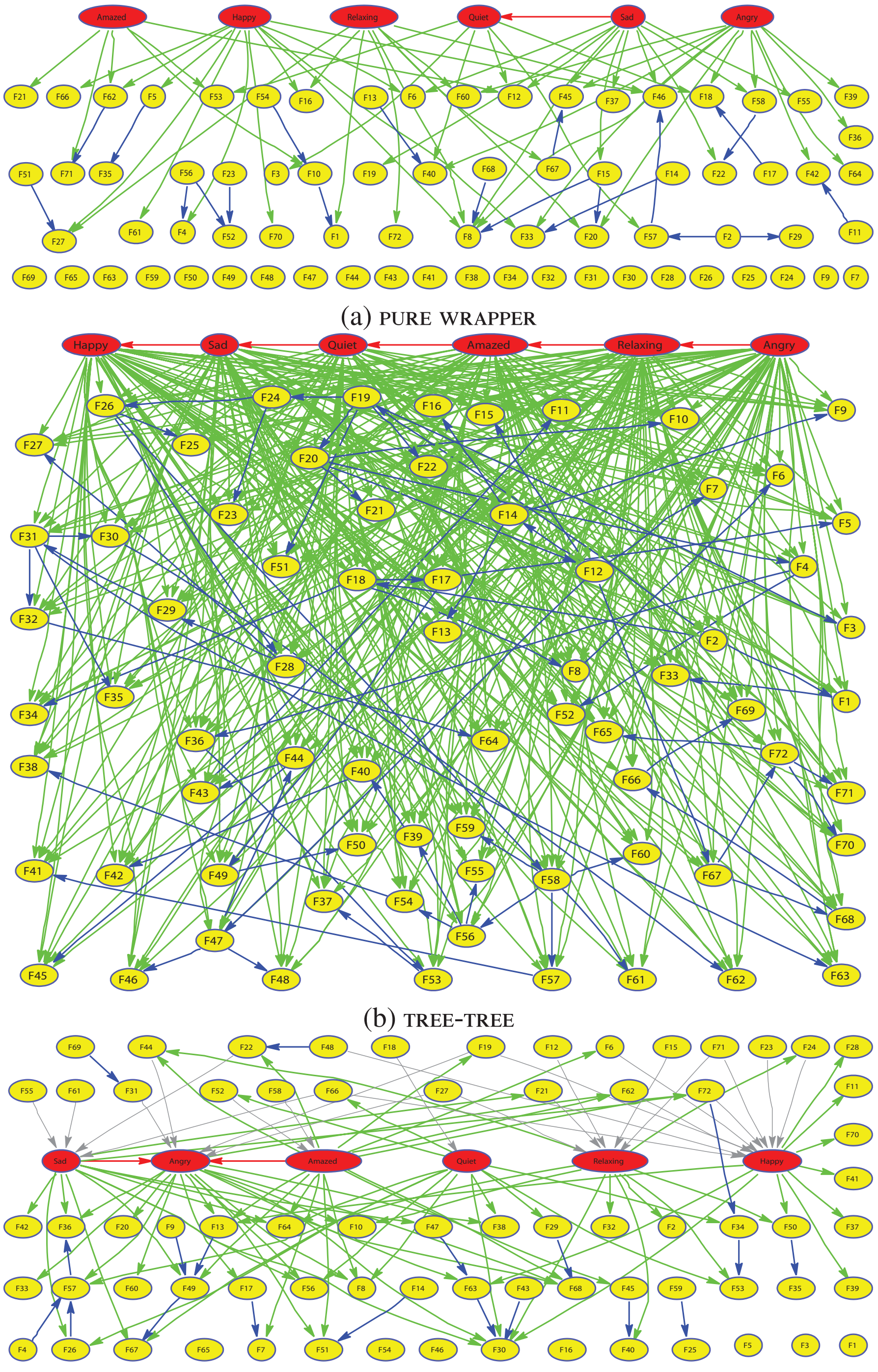
All the experiments have been run on an Intel Core 2, running at 2.40GHz, with 3.5GB RAM using Linux operating system and Matlab parallel programming.

9. Related Work

In this section we review works proposed for approaching multi-dimensional classification problems. The review is organized into four subsections, discussing research addressing multi-label classification, structured prediction, multiple fault diagnosis with Bayesian networks and multi-dimensional Bayesian networks classifiers, respectively. In the last section, containing works quite closely related to our proposal, we stress the similarities and dissimilarities between our research and the state-of-the-art.

9.1 Multi-Label Classification

Multi-label learning has recently originated from modern applications like text and music categorization, protein function and semantic scene classification, where each instance is associated with a subset of labels (present in the instance) from a set of d labels. Therefore, the cardinality of


Amazed
Happy
Relaxing
Quiet
Sad
Angry
F21 F66 F62 F5 F53 F54 F16 F13 F6 F60 F12 F45 F37 F46 F18 F58 F55 F39
F36
F51 F71 F35 F56 F23 F3 F10 F19 F40 F68 F67 F15 F14 F22 F17 F42 F64
F27 F61 F4 F52 F70 F1 F72 F8 F33 F20 F57 F2 F29 F11
F69 F65 F63 F59 F50 F49 F48 F47 F44 F43 F41 F38 F34 F32 F31 F30 F28 F26 F25 F24 F9 F7
Happy
Sad
Quiet
Amazed
Relaxing
Angry
F24 F19 F26 F16 F15 F11 F10 F9 F27 F25 F20 F22 F7 F6 F23 F21 F14 F31 F30 F5 F51 F12 F4 F18 F17 F32 F29 F13 F2 F3 F28 F8 F33 F35 F34 F1 F69 F52 F65 F36 F64 F72 F44 F38 F40 F43 F66 F71 F59 F39 F50 F60 F70 F41 F42 F49 F55 F58 F67 F37 F54 F68 F56 F47 F45 F46 F48 F53 F57 F61 F62 F63
F69 F44 F22 F48 F18 F19 F12 F6 F15 F71 F23 F24 F28
F55 F61 F31 F52 F58 F66 F27 F21 F62 F72 F11
F70
Sad
Angry
Amazed
Quiet
Relaxing
Happy
F41
F42 F36 F20 F9 F13 F64 F10 F47 F38 F29 F32 F2 F34 F50 F37
F33 F57 F60 F49 F17 F56 F8 F14 F63 F43 F68 F45 F59 F53 F35 F39
F4 F26 F67 F65 F7 F51 F54 F46 F30 F16 F40 F25 F5 F3 F1

(a) PURE WRAPPER

(b) TREE-TREE

(c) WRAPPER BN

Figure 7: Graphical structures learnt with the best algorithms for the Emotions data set

this subset varies depending on the instance. The aim is to build a model able to predict the subset of labels for each unseen instance. The problem can be cast into a multi-dimensional classification

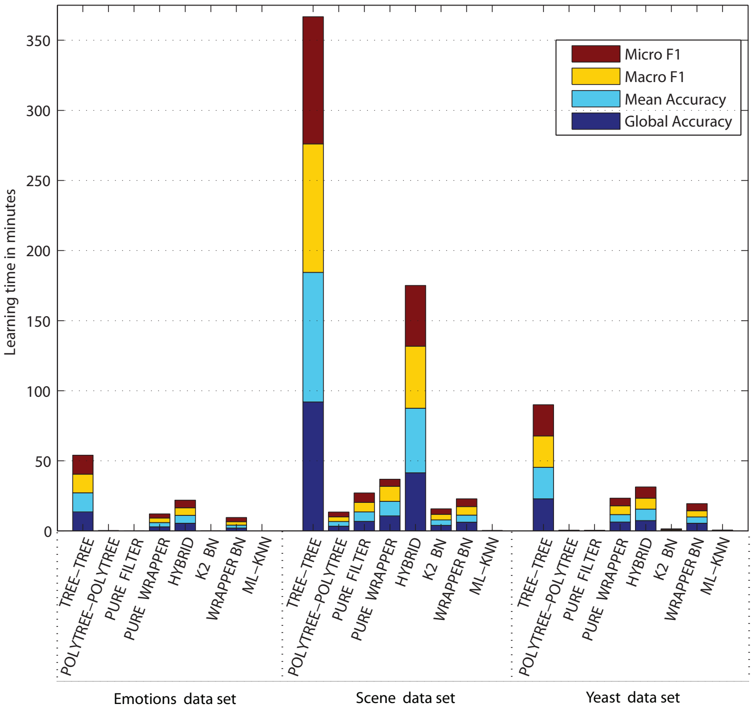


Figure 8: Computation times of the learning process with the eight algorithms when using each performance metric for each data set

problem where all class variables are binary. The particular semantics of the problem requires the use of different metrics than are used in traditional single-label classification, like Hamming loss, one-error, coverage, ranking loss and average precision.

An overview of multi-label classification is given in Tsoumakas and Katakis (2007), where two main categories are distinguished: (a) problem transformation methods, and (b) algorithm adaptation methods. Methods in (a) transform the multi-label classification problem into either one or more single-label classification problems. Methods in (b) extend specific learning algorithms to handle multi-label data directly. For example, decision trees (Vens et al., 2008), support vector machines (Boutell et al., 2004), k-nearest neighbor (Zhang and Zhou, 2007), neural networks (Zhang and Zhou, 2006), and a hybrid of logistic regression and k-nearest neighbor (Cheng and Hüllermeier, 2009) have been proposed.

9.2 Structured Prediction

Structured prediction is a framework for solving classification and regression problems in which the output variables are constrained. The output space consists of structured objects, such as sequences, strings, trees, lattices or graphs. Applications range from machine translation, natural language parsing, object segmentation, stereo reconstruction, human pose estimation, natural scene analysis and protein alignment. The aim is to learn functional dependencies for these complex output spaces, i.e. to compute the structure that maximizes some function of an input parameterized by a weight vector. Methods are based on likelihood like conditional random fields (Sutton et al., 2007), on max-margin (Taskar et al., 2003; Tsochantaridis et al., 2005), and on search (Daumé III and Marcu, 2005).

The state-of-the-art of structured prediction is given in (Bakir et al., 2007) and in a recent special issue (Parker et al., 2009). Challenges addressed currently are related to making the underlying optimization problem more efficient (Hsu et al., 2009), to incorporating prior domain knowledge (from an expert or from auxiliary data) into learning (Mao and Lebanon, 2009), and to dealing with label noise (Lampert and Blaschko, 2009). Finally, Sutton and McCallum (2009) propose breaking complex graphs into tractable pieces which are trained separately. This is called piecewise training, and potentially bears some resemblance to our decomposable MBCs.

9.3 Multiple Fault Diagnosis with Bayesian Networks

A related field is the diagnosis of systems with multiple faults. These systems are devices composed of components (class variables) that can be in either good or failing condition and there are some input/output variables (feature variables) related to the system function. The aim is to find the failing component, or the set of failing components, that explains the observed breakdown. From a probabilistic viewpoint this problem is equivalent to Equation (2), a multi-dimensional classification problem where, as in multi-label classification, all class variables are binary. This is a difficult problem due to the reliability of such systems, where very few breakdown scenarios have been recorded.

Bayesian networks have been used in this context. However, most researchers build these networks systematically by taking advantage of the logical relationships among the variables included in the (physical) devices. A consequence is that only causal networks, a special case of Bayesian networks, are used (Peng and Reggia, 1987a,b). Moreover, the structure and conditional probabilities are not learnt from data but from experts or by supplied specifications. This is the case of Darwiche (1995) and Ibargüengoytia et al. (2000) for example. Neither of them compute the posterior probabilities of the multiple diagnosis. In contrast, Delcroix et al. (2007) try to do this, but by assuming a hypothesis of strong independence among class variables. Elementary Bayesian networks that model each component, following the process described by Geffner and Pearl (1987), are then assembled to have the overall system structure, without connections among class variables. This and other simplifying hypotheses on some conditional probabilities are useful for approximating the posterior probability of a multiple diagnosis. Lucas (2001) combines logic and probability to avoid inconsistencies thereby reducing the complexity of multiple diagnosis computations.

Even simpler approaches, because they all search for single (one component) rather than multiple diagnosis, are Breese and Heckerman (1996); Jensen et al. (2001) for troubleshooting systems; Kipersztok and Dildy (2002) for airplanes; and Spiegelhalter et al. (1993) for congenital heart disease.

9.4 Multi-Dimensional Bayesian Networks Classifiers

To the best of our knowledge, there are four papers that are quite closely related to our proposal: Qazi et al. (2007), van der Gaag and de Waal (2006), de Waal and van der Gaag (2007) and Rodríguez and Lozano (2008). They all learn the structure from data constrained to a pre-set family of MBCs.

Qazi et al. (2007) propose learning a `DAG-empty` MBC to predict heart wall motion for the 16 segments (class variables) of the heart. Once the DAG for the class subgraph is learnt by standard Bayesian networks procedures, a naïve Bayes model containing a subset of feature variables is obtained for each class variable using different features for each naïve Bayes model to finally build the corresponding bridge subgraph.

Van der Gaag and de Waal (2006) use `tree-tree` MBCs as follows. They prove that the score+search learning strategy based on the MDL score can be decomposed into optimization problems for the set of class variables and for the set of feature variables separately. The class subgraph is firstly learnt by searching for the maximum weighted undirected spanning tree and transforming it into a directed tree using Chow and Liu's algorithm (Chow and Liu, 1968). The weight of an edge is the mutual information between a pair of class variables. For a fixed bridge subgraph, the feature subgraph is then learnt by building a maximum weighted directed spanning tree (Chu and Liu, 1968). The weight of an arc is the conditional mutual information between pairs of feature variables given the parents (classes) of the second feature determined by the bridge subgraph. Unlike these two filter learning processes, the bridge subgraph is greedily changed in a wrapper-like way, trying to improve the accuracy Acc as defined in Equation (7). Acc is the measure used to assess the quality of the learnt `tree-tree` MBC.

De Waal and van der Gaag (2007) theoretically find the conditions for the optimal recovery of polytree structures in class and features subgraphs for the case of `polytree-polytree` MBCs. The algorithms for learning the polytrees for the class and feature subgraphs separately are based on Rebane and Pearl's algorithm (Rebane and Pearl, 1987), although for the feature subgraph the algorithm requires considering all the class variables in the conditional mutual information, which is unfeasible in real applications. This theoretical work does not state how the `polytree-polytree` MBC is learnt because it omits how to find the bridge subgraph.

Rodríguez and Lozano (2008) extend polytrees to k-DB structures for class and features subgraphs (a special `DAG-DAG` MBC). Each permitted structure is coded as an individual in a genetic algorithm with three substrings, one per subgraph. The objective function is a vector with the accuracies Acc_j (see Equation (6)) as components. Comparing MBCs amounts to finding non-dominated structures according to this multi-objective fitness function, NSGA-II (Deb et al., 2000) being the chosen multi-objective genetic algorithm.

Unlike these approaches, our proposal defines a unified framework allowing any Bayesian network structure in the three subgraphs of an MBC. CB-decomposable MBCs in conjunction with a gray code for enumerating the joint configurations of all the class variables in a special order, alleviate the computational burden when calculating the MPE. The framework has been extended beyond the 0-1 loss function, to general loss functions, where the additive CB-decomposable functions exploit the structure decomposition to the utmost. While the learning approaches of the MBCs are filter-based for Qazi et al. (2007), hybrid for van der Gaag and de Waal (2006) and wrapper for Rodríguez and Lozano (2008), our framework allows the three strategy types. Also, we have introduced several performance evaluation metrics of the MBC models, useful for wrapper-based algorithms as well as for the quality assessment of the final model. Finally, note that the data sets tested in two of these papers are relatively small ($d = 3$ and $m = 40$ in van der Gaag and de Waal (2006), $d = 2$ and $m = 58$ in Rodríguez and Lozano (2008)) compared with our experiments. A similar size is used in Qazi et al. (2007) ($d = 16$ and $m = 216$), although their structure does not permit dependencies between the feature variables (96 in the final model). De Waal and van der Gaag (2007) do not present any experimental results.

10. Conclusions and Future Research

This paper approaches the multi-dimensional classification problem using a type of probabilistic graphical models named multi-dimensional Bayesian network classifiers. In MBCs, the multi-dimensional classification problem is equivalent, for a 0-1 loss function, to the search for the most probable explanation, which has shown to be a NP-hard problem.

We introduce a new type of MBCs, the so-called class-bridge (CB) decomposable MBCs that alleviates the computational burden for computing MPEs. Also, thanks to an adaptation of the gray code, we can reduce this complexity in both general MBCs and CB-decomposable MBCs even further. Upper bounds for the reductions are obtained for both types of models. Theoretical results on how to obtain the Bayes decision rule for general MBCs in the case of 0-1 loss functions and in the case of additive loss functions for CB-decomposable MBCs are proved. The paper also extends some usual performance evaluation measures, previously defined for the single-class domain, to this multi-dimensional setting. These are accuracy, sensitivity, specificity and F1 measure. Finally, flexible algorithms for learning MBCs from data are shown. Flexibility refers to the different families of permitted MBCs (e.g. `tree-tree`, `polytree-polytree`, `DAG-DAG`, `DAG-polytree`, etc.), as well as to the filter, wrapper and hybrid approaches considered to carry out the learning process. We also provide theoretical results counting the number of all possible MBC structures, that is, the cardinality of the search space for the learning task. Empirical results on the application of these learning algorithms to three data sets taken from the literature on multi-label classification problems are encouraging, beating a state-of-the-art algorithm in this multi-label setting.

This work can be extended and improved in several ways. The consideration of problems when the vector of variables to be predicted includes discrete, as well as continuous variables (classification and regression problems), is a line for a possible extension of the current approach. To avoid the discretization of the feature variables any other possibility for generalizing this paper would be to allow discrete and continuous variables in the feature subgraph.

A characteristic of real-world data sets is the inclusion of instances with missing data. Thus, the adaptation of the EM algorithm to the two-layer architecture of MBCs (or to the more sophisticated CB-decomposable MBC structures) is another line for future research.

We also intend to look at the definition of concept drift for multi-dimensional classification problems in data stream scenarios, and the development of the respective detection procedures for MBCs in the future.

Acknowledgments

Research partially supported by the Spanish Ministry of Science and Innovation (projects TIN2007-62626, TIN2008-06815-C02-02, Consolider Ingenio 2010-CSD2007-00018 and Cajal Blue Brain) and by FONCICYT (Dynamo project, UPM-P10 1015-100). The second author was supported by UPM Sino-Spanish Research Scholarship Program.

References

A. M. Abdelbar and S. M. Hedetniemi. Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, 102(1):21–38, 1998.

G. H. Bakir, T. Hofmann, B. Schölkopf, A.J. Smola, and B. Taskar. *Predicting Structured Data*. Cambridge: MIT Press, 2007.

R. Blanco, I. Inza, M. Merino, J. Quiroga, and P. Larrañaga. Feature selection in Bayesian classifiers for the prognosis of survival of cirrhotic patients treated with TIPS. *Journal of Biomedical Informatics*, 38(5):376–388, 2005.

M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

J. S. Breese and D. Heckerman. Decision-theoretic troubleshooting: A framework for repair and experiment. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence*, pages 124–132, 1996.

W. Cheng and E. Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.

T. Cheng-Jung, L. Chien-I, and Y. Wei-Pang. A discretization algorithm based on class-attribute contingency coefficient. *Information Sciences*, 178:714–731, 2008.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

C. Chu and T.H. Liu. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400, 1968.

G. F. Cooper and E. A. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

A. Darwiche. Model-based diagnosis using causal networks. In *International Joint Conference on Artificial Intelligence*, pages 211–219, 1995.

H. Daumé III and D. Marcu. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the International Conference on Machine Learning*, pages 169–176, 2005.

A. P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2:25–36, 1992.

P. R. de Waal and L. C. van der Gaag. Inference and learning in multi-dimensional Bayesian network classifiers. In *European Conference on Symbolic and Quantitative Approaches to Reasoning under Uncertainty, Lecture Notes in Artificial Intelligence*, volume 4724, pages 501–511. Springer, 2007.

K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In *Parallel Problem Solving from Nature (PPSN VI). Lecture Notes in Computer Science, 1917*, pages 849–858, 2000.

R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113 (1-2):41–85, 1999.

R. Dechter and I. Rish. A scheme for approximating probabilistic inference. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 132–141, 1997.

V. Delcroix, M.-A. Maalej, and S. Piechowiak. Bayesian networks versus other probabilistic models for the multiple diagnosis of large devices. *International Journal on Artificial Intelligence Tools*, 16(3):417–433, 2007.

R. O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification (2nd Edition).* Wiley-Interscience, 2000.

A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, pages 681–687. MIT Press, 2002.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29: 131–163, 1997.

S. García and F. Herrera. An extension on “Statistical comparisons of classifiers over multiple data sets” for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

H. Geffner and J. Pearl. An improved constraint-propagation algorithm for diagnosis. In *International Joint Conference on Artificial Intelligence*, pages 1105–1111, 1987.

E. S. Gelsema. Abductive reasoning in Bayesian belief networks using a genetic algorithm. *Pattern Recognition Letters*, 16(8):865–871, 1995.

D.-J. Guan. Generalized gray codes with applications. *Proceedings of the National Science Council of the Republic of China (A)*, 22(6):842–848, 1998.

C.-N. Hsu, H.-S. Huang, and Y.-M. Chang. Periodic step-size adaptation in second-order gradient descent for single-pass on-line structured learning. *Machine Learning*, 77(2-3):195–224, 2009.

F. Hutter, H. H. Hoos, and T. Stützle. Efficient stochastic local search for MPE solving. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 169–174, 2005.

P. Ibargüengoytia, L. E. Sucar, and E. Morales. Probabilistic model-based diagnosis. In *Proceedings of MICAI 2000, Lecture Notes in Artificial Intelligence*, volume 1793, pages 687–698, 2000.

F.V. Jensen, U. Kjærulff, B. Kristiansen, H. Langseth, C. Skaanning, J. Vomlel, and M. Vomlelová. The SACSO methodology for troubleshooting complex systems. *Artificial Intelligence for Engineering Design Analysis and Manufacturing*, 15(4):321–333, 2001.

K. Kask and R. Dechter. A general scheme for automatic generation of search heuristics from specification dependencies. *Artificial Intelligence*, 129(1-2):91–131, 2001.

K. Kask and R. Dechter. Mini-bucket heuristics for improved search. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 314–323, 1999.

O. Kipersztok and G.A. Dildy. Evidence-based Bayesian networks approach to airplane maintenance. In *Proceedings of the 2002 International Joint Conference on Neural Networks*, volume 3, pages 2887–2891, 2002.

R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2): 273–324, 1997.

D. Koller and N. Friedman. *Probabilistic Graphical Models. Principles and Techniques*. Cambridge: MIT Press, 2009.

C. H. Lampert and M. B. Blaschko. Structured prediction by joint kernel support estimation. *Machine Learning*, 77(2-3):249–269, 2009.

P. Langley and S. Sage. Tractable average-case analysis of naïve Bayesian classifiers. In *Proceedings of the 16th International Conference on Machine Learning*, pages 220–228, 1999.

Z. Li and B. D'Ambrosio. An efficient approach for finding the MPE in belief networks. In *Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence*, pages 342–349, 1993.

P. Lucas. Bayesian model-based diagnosis. *International Journal of Approximate Reasoning*, 27 (2):99–119, 2001.

Y. Mao and G. Lebanon. Generalized isotonic conditional random fields. *Machine Learning*, 77 (2-3):225–248, 2009.

R. Marinescu and R. Dechter. AND/OR branch-and-bound search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17):1457–1491, 2009.

M. Minsky. Steps toward artificial intelligence. *Transactions on Institute of Radio Engineers*, 49: 8–30, 1961.

J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.

C. Parker, Y. Altun, and P. Tadepalli. Guest editorial: Special issue on structured prediction. *Machine Learning*, 77:161–164, 2009.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

Y. Peng and J. A. Reggia. A probabilistic causal model for diagnostic problem solving– Part I: Integrating symbolic causal inference with numeric probabilistic inference. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(2):146–162, 1987a.

Y. Peng and J. A. Reggia. A probabilistic causal model for diagnostic problem solving. Part II: Diagnostic strategy. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(3):395–406, 1987b.

M. Qazi, G. Fung, S. Krishnan, R. Rosales, H. Steck, R.B. Rao, D. Poldermans, and D. Chandrasekaran. Automated heart wall motion abnormality detection from ultrasound images using Bayesian networks. In *International Joint Conference on Artificial Intelligence*, pages 519–525, 2007.

G. Rebane and J. Pearl. The recovery of causal polytrees from statistical data. In *Proceedings of the Third Annual Conference on Uncertainty in Artificial Intelligence (UAI-87)*, volume 3, pages 222–228. Elsevier, 1987.

R. W. Robinson. Counting labeled acyclic digraphs. In *New Directions in Graph Theory*. New York: Academic Press, 1973.

J. D. Rodríguez and J. A. Lozano. Multi-objective learning of multi-dimensional Bayesian classifiers. In *Proceedings of the Eighth International Conference on Hybrid Intelligent Systems*, pages 501–506, 2008.

C. Rojas-Guzmán and M. A. Kramer. GALGO: A genetic algorithm decision support tool for complex uncertain systems modeled with Bayesian belief networks. In *Proceedings of the Ninth Annual Conference on Uncertainty in Artificial Intelligence*, pages 368–375, 1993.

M. Sahami. Learning limited dependence Bayesian classifiers. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 335–338, 1996.

E. Santos. On the generation of alternative explanations with implications for belief revision. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 339–347, 1991.

S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.

S. E. Shimony and E. Charniak. A new algorithm for finding MAP assignments to belief networks. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 185–196, 1990.

D. J. Spiegelhalter, A. P. Dawid, S. L. Lauritzen, and R. G. Cowell. Bayesian analysis in expert systems. *Statistical Science*, 8(3):219–247, 1993.

M. Stone. Cross-validatory choice and assessment of statistical predictions (with discussion). *Journal of the Royal Statistical Society B*, 36:111–147, 1974.

C. Sutton and A. McCallum. Piecewise training for structured prediction. *Machine Learning*, 77 (2-3):165–194, 2009.

C. Sutton, A. McCallum, and K. Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labelling and segmenting sequence data. *Journal of Machine Learning Research*, 8:693–723, 2007.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Neural Information and Processing Systems (NIPS)*, 2003.

K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas. Multi-label classification of music into emotions. In *International Society for Music Information Retrieval Conference*, pages 325–330, 2008.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007.

L.C. van der Gaag and P. R. de Waal. Multi-dimensional Bayesian network classifiers. In *Third European Conference on Probabilistic Graphical Models*, pages 107–114, 2006.

C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008.

M. L. Zhang and Z. H. Zhou. Multi-label neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006.

M.L. Zhang and Z.H. Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.