A scalable pairwise class interaction framework for multidimensional classification☆

Jacinto Arias[a,*], Jose A. Gamez[a], Thomas D. Nielsen[b], Jose M. Puerta[a]

[a] *Department of Computing Systems, University of Castilla-La Mancha, Albacete, Spain*
[b] *Department of Computer Science, Aalborg University, Denmark*

Abstract

We present a general framework for multidimensional classification that captures the pairwise interactions between class variables. The pairwise class interactions are encoded using a collection of base classifiers (Phase 1), for which the class predictions are combined in a Markov random field that is subsequently used for multidimensional inference (Phase 2); thus, the framework can be positioned between multilabel Bayesian classifiers and label transformation-based approaches. Our proposal leads to a general framework supporting a wide range of base classifiers in the first phase as well as different inference methods in the second phase. We describe the basic framework and its main properties, as well as strategies for ensuring the scalability of the framework. We include a detailed experimental evaluation based on a range of publicly available databases. Here we analyze the overall performance of the framework and we test the behavior of the different scalability strategies proposed. A comparison with other state-of-the-art multidimensional classifiers show that the proposed framework either outperforms or is competitive with the tested straw-men methods.

Keywords: Multidimensional classification, probabilistic classifiers, Markov random fields

1. Introduction

Supervised classification is the problem of assigning a value to a distinguished variable, the *class* C, for a given instance defined over a set of predictive attributes. In *multi-label* classification, several class variables are simultaneously considered and the task consists of assigning a configuration of values to all the class variables. In the multi-label setting, classes (or *labels*) are binary. *Multi-dimensional* classification is a generalization of multi-label classification that allows class variables to have more than two values [2]. Recent literature,

☆This paper is an extension of the conference paper [1].

*Corresponding author
Email addresses: jacinto.arias@uclm.es (Jacinto Arias), jose.gamez@uclm.es (Jose A. Gamez), tdn@cs.aau.dk (Thomas D. Nielsen), jose.puerta@uclm.es (Jose M. Puerta)

however, also uses the term multi-label classification when dealing with n-ary class variables, so we will use both names in an interchangeable way.

A wide range of applications for multi-dimensional classification has been found [3, Section 1]: bio-informatics, document/music/movie categorization, semantic scene classification, multi-fault diagnosis, etc. One approach to solve a multi-dimensional problem is to first *transform* the problem into a set of single-class classification problems, and then combine the outputs to obtain a joint configuration of the class values. Transformation-based methods range from binary relevance, where no interaction among the class variables is modeled, to brute-force label power set methods, where all the class variables are aggregated into a single compound class. In-between these two extremes, we find new and/or adapted algorithms that have been developed to deal with the multi-dimensional problem, managing in a natural way the interactions between the variables. From this second family, probabilistic methods and, in particular, those based on Bayesian networks (BNs) [4] have demonstrated a convincing performance [5]. In this paper we focus on the probabilistic approach to multi-dimensional classification.

We propose a two-stage framework for multi-dimensional classification. The framework can be positioned between the transformation-based classifiers and the family of multi-dimensional probabilistic graphical models (PGMs)-based classifiers[1]. In the first stage we learn a single-class classifier for each pair of class variables in the domain, hence this stage of the framework follows a transformation-based approach. The framework does not prescribe a particular type of classifier, but only requires that the outcome of the classifier should be a weighted distribution over the (compound) class values. Standard probabilistic classifiers meet this criterion. In the second stage a Markov random field (MRF) is constructed based on the results from the first stage. The MRF thus models the dependencies between the class variables, and thereby connects the framework to the class of multi-dimensional PGM-based classifiers. Subsequent classification is achieved by performing inference in the induced MRF.

The proposed framework is flexible: (1) different types of classifiers can be applied in the first stage; (2) preprocessing can be done separately for each single-class classifier, thus allowing one to take advantage of state-of-the-art algorithms for supervised discretization and feature selection; and (3) different types of MRF-based inference algorithms can be used for the subsequent classification, and the choice of method can therefore depend on the complexity of the model (exact or approximate inference) and the score to be maximized (calculation of marginal probabilities or a most probable explanation). Furthermore, the general method scale with the computational resources available as the base classifiers in the first stage can be learned independently and chosen to also support scalability at the individual classifier level [7]. Nevertheless, naively dealing with all pairs of class variables imposes a strong limitation on the number of class vari-

[1] We refer to PGM based classifiers in order to accommodate a wider range of graphical models [6] in addition to the more common approaches based on Bayesian networks.

ables the algorithm can handle. We therefore also outline strategies for scaling up the algorithm to datasets having a large number of class variables. Experiments carried out over a collection of benchmark datasets confirm the feasibility of the approach and show that the proposed method significantly outperforms or is comparable to the straw-men methods included in the comparison. This paper has a companion website, where the source-code and additional experimental results can be found: `http://simd.albacete.org/supplements/FMC.html`.

We would like to remark that the proposal does not fit into the so-called *pair-wise multi-label approach*, which interprets the labels of the instances as preferences and whose goal is to obtain a *ranking* among the labels [8] and not a joint configuration of class values. Furthermore, although our approach trains several classifiers in the first phase, it does not strictly fall into the class of *ensemble* methods either, as each base-classifier only provides a partial answer to the multi-dimensional problem.

This paper extends a previous preliminary study [1]. Specifically, we explore additional strategies to ensure scalability of the framework and we provide a significantly expanded experimental analysis and comparison with other state-of-the-art straw-men methods covering both accuracy and run-time performance.

The rest of the paper is structured as follows: In Section 2 we introduce the notation and the required background in the field of multidimensional classification; in Section 3 we describe our framework and discuss its main properties; in Section 4 we introduce and discuss several improvements to the framework with regards to scalability; in Section 5 we evaluate our approach by carrying out several experiments with real world data, and finally, in Section 6, we summarize the obtained results and introduce ideas for future work.

2. Background

2.1. Notation and Problem Definition

We assume that the available dataset consists of a collection of instances $\mathbf{D} = \{(\mathbf{a}^{(1)}, \mathbf{c}^{(1)}), \ldots, (\mathbf{a}^{(t)}, \mathbf{c}^{(t)})\}$, where the first part of an instance, $\mathbf{a}^{(i)} = (a_1^{(i)}, \ldots, a_n^{(i)})$, is a configuration of values defined over a set $\mathbf{A} = \{A_1, \ldots, A_n\}$ of predictive attributes, while the second part, $\mathbf{c}^{(i)} = (c_1^{(i)}, \ldots, c_m^{(i)})$, is a configuration of values defined over a set $\mathbf{C} = \{C_1, \ldots, C_m\}$ of classes.[2] Due to the restrictions of the models analyzed in this paper, we will assume that all the variables are discrete (nominal), i.e., the state spaces $dom(A_i)$ and $dom(C_j)$ are finite sets of mutually exclusive and exhaustive states, $\forall i, 1 \leqslant i \leqslant n$ and $\forall j, 1 \leqslant j \leqslant m$. In Section 3.1 we will discuss the preprocessing capabilities of our proposal in relation to continuous data.

[2] We will omit the superscript when no confusion is possible, just writing $(\mathbf{a}, \mathbf{c})$ and $((a_1, \ldots, a_n), (c_1, \ldots, c_m))$ instead of $(\mathbf{a}^{(i)}, \mathbf{c}^{(i)})$ and $((a_1^{(i)}, \ldots, a_n^{(i)}), (c_1^{(i)}, \ldots, c_m^{(i)}))$, respectively.

Our goal is to induce a multidimensional classifier f that maps configurations of the predictive variables to configurations of the class variables:

$$f \ : \ \times_{i=1}^{n} dom(A_i) \longrightarrow \times_{j=1}^{m} dom(C_j),$$

$$(x_1, x_2, \ldots, x_n) \mapsto (c_1, c_2, \ldots, c_m),$$

where $\times$ denotes the Cartesian product.

2.2. Evaluation

Although different evaluation metrics can be used to evaluate a multidimensional classifier (see e.g. [5, Sec. 5]), we resort in this paper to two of the more widely used ones, both relating to *accuracy*. Given a dataset $\mathbf{D}$ consisting of t multidimensional instances $((a_1, \ldots, a_n), (c_1, \ldots, c_m))$ together with the predictions obtained by a multidimensional classifier $f(a_1, \ldots, a_n) = (c'_1, \ldots, c'_m)$, we employ the following evaluation functions:

- *Exact match* (or *global accuracy*).

$$acc(\mathbf{D}, F) = \frac{1}{t} \sum_{i=1}^{t} \delta(\mathbf{c}^{(i)}, \mathbf{c}'^{(i)}), \tag{1}$$

  where δ is Kronecker's delta function.

- *Hamming* (or *mean*) *accuracy*.

$$H_{acc}(\mathbf{D}, F) = \frac{1}{t} \sum_{i=1}^{t} \frac{1}{m} \sum_{j=1}^{m} \delta(c_j^{(i)}, c_j'^{(i)}). \tag{2}$$

In both cases, the higher the value the better. Obviously, *acc* is a harder scoring criterion than H_{acc}.

From a probabilistic perspective, different inference tasks [4] can be used in order to maximize each of the two scores. Thus, computing the *most probable explanation* (MPE) for the class variables will maximize global accuracy, while maximizing the Hamming accuracy requires the computation of the most probable marginal assignment for each of the individual class variables.

2.3. Approaches to Multi-Dimensional Classification

In the literature we can find several approaches dealing with multi-dimensional classification problems. Below, we briefly review some of them, detailing a bit more those approaches used for comparison in this paper and those that are more related to our proposal. A more general and extensive overview and comparison of existing algorithms can be found in [3] and [9], respectively.

Transformation methods. According to [10] this family of methods transform the multi-dimensional classification problem into one or several single-class classification problems. Perhaps the two most well-known approaches coming from

the multi-label domain are the classifiers based on *label power-sets* (LP) and *binary relevance* (BR). In their simplest form, LP-based classifiers construct a new (compound) single-class variable having as possible values all the different configurations of the class values (labels) included in the training set. This method implicitly considers the dependencies between classes, but its obvious main drawback is that it is computationally tractable only for a relatively small number of class variables.

On the other hand, *Binary Relevance* (BR) methods learn a single-class classifier (or base classifier) for each class variable, $f_i : \times_{j=1}^{n} dom(A_j) \rightarrow dom(C_i)$. A solution to the multi-dimensional problem is then found by combining the single-class outputs from the base classifiers. This method does not consider any dependencies between class variables, and it usually performs poorly when considering exact accuracy. However it can still provide good predictions according to Hamming accuracy for certain domains.

In-between BR and brute-force LP-based methods other approaches have been developed. One example is RAkEL [11], which is based on training several single-class classifiers each using as class a compound variable constructed as the Cartesian product of k class variables. The number of single-class variable models is usually linear in the number of class variables, and random selection is used to choose the k variables that will form the compound variable. Later, from the $k-$tuples predicted, voting is used to obtain a joint configuration over all the class variables.

Chain classifiers (CC) [12] are an alternative to BR that incorporate dependencies between class variables, while still maintaining the computational efficiency of BR. In CC an ordering σ is defined over the class variables. Let $C_{\sigma(i)}$ be the i-th class variable according to ordering σ. As in BR, m single-class classifiers are induced in a CC, but when learning the single-class classifier having $C_{\sigma(i)}$ as class, the variables $C_{\sigma(1)}, \ldots, C_{\sigma(i-1)}$ are also included as predictive attributes: $f_i : \times_{j=1}^{n} dom(A_j) \times \times_{k=1}^{i-1} dom(C_{\sigma(k)}) \rightarrow dom(C_{\sigma(i)})$. Therefore, the class variable in position i of σ, depends on the class variables appearing earlier in the ordering. As a consequence, inference over the single-class classifiers must be done sequentially by following the ordering imposed by σ and by using the predicted values of the previous class variables as input when performing inference. This ordering restriction is lifted in [13, 14], where each single class classifier uses all the other class variables as predictive attributes; during inference, the values of these additional input features are found using a separate set of base classifiers. A probabilistic approach to chain classifiers was proposed in [15], which generalizes the original chain classifier by letting the single class classifiers f_i define a probability distribution over $C_{\sigma(i)}$. In combination, the single class classifiers define a joint distribution over all class variables, for which the original chain classifier can be seen as a deterministic approximation. The cost of this generalization is a corresponding increase in model complexity, which necessitates the use of approximate inference algorithms for domains with even a moderate number of class variables.

Adaptation methods. These are methods that directly modify/adapt existing

single-class classification algorithms to accommodate multiple classes, e.g. based on decision trees, nearest neighbors, support vector machines, etc. See [10] for an overview.

Multi-dimensional Bayesian Networks classifiers (MBCs) [16, 17, 5]. These are multi-dimensional classifiers that use the formalism of BNs to model the problem. However, as for single-class domains, instead of learning an unconstrained BN, the learning process is constrained by biasing the resulting graph. Thus, in MBCs three subgraphs are usually considered: (a) the class subgraph, which codifies dependence relations between classes; (b) the feature subgraph, which codifies dependence relations between features (predictive attributes); and (c) the bridge subgraph which codifies dependence relations from classes to features.

Depending on the complexity of the types of graphs allowed in the class and feature subgraphs, several models/algorithms can arise: trees and/or polytrees [16, 17, 18], k-dependence limited models [18], general BN structures [5], etc. With respect to the search strategy used to guide the learning process, filter and wrapper approaches have been analyzed in [5], while skeleton-based ones are proposed in [19, 20] based on Markov blankets and in [21] using mutual information. In [22] a slightly different approach is introduced, based on concentrating the effort in the class and bridge subgraph, while assuming an empty feature subgraph, that is, features are conditionally independent given the classes. For the class subgraph, restricted topology structures (naive Bayes and forest augmented networks) are considered, while for the bridge subgraph an exact score+search method is used that considers all possible parents sets of the features. The accuracy of the obtained classifiers is slightly worse than the ensemble proposed by the same authors [23], but inference is much faster. The method proposed in [23] assume that the input features are conditionally independent given the class labels, hence focus is on learning structures over the class labels. In contrast, [24] considers a mixture of tree structures over the class labels, but here each of the mixture trees are conditioned on the input features.

Ensembles. As in single-class classification, ensembles of multi-dimensional classifiers have shown potential to improve performance compared to single classifiers. This is, for example, the case of the ensemble of CC, where each member of the ensemble uses a different (usually random) ordering [12]. In the particular case of using MBCs as base classifiers for the ensemble, recent studies [25, 26, 23] explore the idea of using as many members in the ensemble as there are class variables. In [25, 26] an undirected tree structure is first learned for the class variables, and then each class variable is set as root and the resulting topological ordering is used as a CC in the ensemble. The resulting configuration of class values is obtained by voting. In [23] no structural learning is done over the classes, and instead a naive Bayes structure among the class variables is used, but with a different root for each member of the ensemble. In contrast to previous approaches, the resulting configuration is obtained by probabilistic inference.

Finally, we would like to mention two recent works which show the degree of maturity research on MBCs has achieved. In [27] a theoretical study is carried out to analyze the expressive power of binary relevance and chain classifiers, proving that chain classifiers provide more expressive models than the binary relevance method when the same type of BAN (BN augmented naive Bayes) classifier is used as the base classifier. On the other hand, [28] addressed the problem of Hierarchical Multi-label Classification (HMC), which addresses domains for which there exists a hierarchical structure among the classes.

3. A General Framework for Multi-Label Classification

In this section we describe the proposed framework for multi-label classification and its main capabilities.

The proposed framework for doing multi-label classification positions itself between the family of MBCs classifiers [16, 17, 5] and transformation-based classifiers [10], by combining the results from a collection of classifiers learned for each possible pairwise interaction between the class variables. The legal types of classifiers are restricted to those classifiers that for a given instance $\mathbf{a}$ can provide a factor $\phi_{ij\,|\,\mathbf{a}} : dom(C_i, C_j) \rightarrow \mathbb{R}^+$ for each class pair C_i and C_j such that the greater the value the higher the dependence between the class states. We shall refer to these classifiers as *base classifiers*. Given the class-pair factors produced by the base classifiers for an instance $\mathbf{a}$, we pose the problem of doing multi-label classification as an inference problem in the pairwise Markov random field (MRF) induced by the factors. Specifically, for m class variables, the pairwise Markov random field specified by the factors $\phi_{ij\,|\,\mathbf{a}}$ defines a joint distribution

$$P(C_1, \ldots, C_m \,|\, \mathbf{a}) = \frac{1}{Z} \prod_{i \neq j} \phi_{ij\,|\,\mathbf{a}}(C_i, C_j),$$

where

$$Z = \sum_{C_1, \ldots, C_m} \prod_{i \neq j} \phi_{ij\,|\,\mathbf{a}}(C_i, C_j)$$

is the partition function.[3] Based on this specification, we perform classification by doing inference in the MRF model. Thus, for global accuracy we look for the most probable explanation (MPE) in the MRF

$$\begin{aligned} \mathbf{c}^* &= \arg \max_{\mathbf{c}=(c_1,\ldots,c_m)} \frac{1}{Z} \prod_{i \neq j} \phi_{ij\,|\,\mathbf{a}}(C_i, C_j) \\ &= \arg \max_{\mathbf{c}=(c_1,\ldots,c_m)} \prod_{i \neq j} \phi_{ij\,|\,\mathbf{a}}(C_i, C_j), \end{aligned}$$

[3] We abuse notation slightly and use variable summation to denote summation over states of a variable.

and for Hamming accuracy we consider the most probable class variable configurations separately:

$$c_k^* = \arg\max_{c_k} \sum_{C_l : l \neq k} \prod_{i \neq j} \phi_{ij \,|\, \mathbf{a}}(C_i, C_j).$$

In summary, given a collection of base classifiers, multi-label classification of an instance $\mathbf{a}$ consists of two steps:

1. For each pair of class variables, C_i and C_j, employ the corresponding base classifier to find a factor $\phi_{ij \,|\, \mathbf{a}}$ that for each configuration (c_i, c_j) encodes the dependence between c_i and c_j for instance $\mathbf{a}$.[4]
2. Establish the class-labels of $\mathbf{a}$ by performing inference in the pairwise Markov random field defined by the factors $\phi_{ij \,|\, \mathbf{a}}$ found in step 1.

The overall framework is flexible in the sense that it can accommodate several different types of base classifiers (e.g., probabilistic classifiers, neural networks, etc.). Hence, we say that a particular choice of base classifier instantiates the framework, and in what follows we shall refer to the instantiated framework as a *factor-based multi-labeled classifier (FMC)*. In the present paper, we focus on probabilistic base classifiers, and for ease of exposition we will often only refer to naive Bayes classifiers even though other base classifiers can be applied: for each pair of class variables C_i and C_j we thus have a naive Bayes classifier (NBC), where the state space of the class variable consists of all combinations of class labels for C_i and C_j. With this type of base classifier, the factors $\phi_{ij \,|\, \mathbf{a}}$ in the FMC correspond to the posterior probabilities $P(C_i, C_j \,|\, \mathbf{a})$. The relationship between the NB base classifiers and the induced MRF is illustrated in Figure 1 for a domain with three attributes $\{A_1, A_2, A_3\}$ and three class variables $\{C_1, C_2, C_3\}$.

Comparing the proposed framework to multi-dimensional Bayesian network classifiers (MBCs) [17, 5, 19] we see that the induced MRF plays the role of the class-subgraph in the MBC. Similarly, the feature subgraph and the bridge subgraph are captured by the base classifiers that, in addition, also allow each class pair to employ different types of preprocessing and encode different dependency structures. The proposed method also shares some similarities with the method by [2], which performs multi-dimensional classification by constructing super-classes based on a partitioning of the class variables. This partitioning is based on (indirect) measurements of conditional class dependencies [29], and for each identified subset a distinct multi-dimensional classifier is learned. A consequence of the class partitioning is that class variables in different super-classes are assumed independent.[5] In comparison, dependencies between class labels in

[4] Clearly, considering all pairs of base classifiers can be computationally intensive, and strategies for learning base classifiers for only a subset of the possible class variable pairs is explored in Section 4.

[5] To make the method more robust to variations in the training data, in particular in relation to the identified dependency structures/class partitioning, [2] also considers ensembles of super-class classifiers.

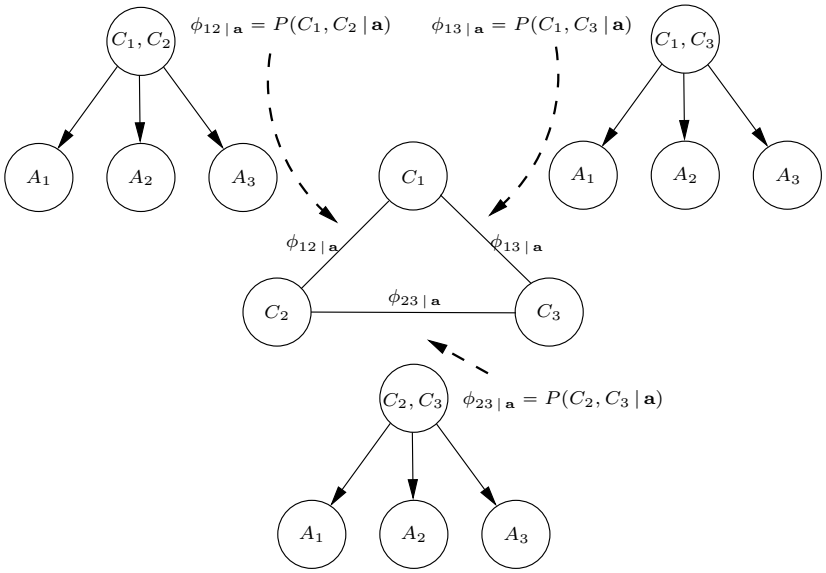

Figure 1: An FMC is established for a domain consisting of three attributes and three class variables. The joint probabilities calculated using the NB base classifiers serve as factors in the MRF, which is in turn used for finding the class labels.

the FMC are encoded in the Markov random field, which is directly obtained based on pair-wise class interactions and provides a flexible way of encoding label dependence without having to assume (marginal) independences between class labels. In the FMC framework, the Markov random field thus has the role of a secondary structure used for combining pairwise class label predictions. A related approach is explored in [30], where a Bayesian network structure is used for capturing label dependencies. The Bayesian network is learned directly from the class labels in the data set, and is subsequently used to combine previously obtained class label estimates (found using existing multi-label classifiers).

3.1. Preprocessing capabilities

Learning an FMC is, in principle, equivalent to independently learning base classifiers for selected class variable pairs. Not only does this support scalable learning methods to be devised, but it also allows for different preprocessing techniques to be deployed for the different base classifiers; this e.g. includes discretization of continuous variables as well as feature subset selection (FSS). Consequently, the discretization of a continuous feature variable A_i appearing in two different base classifiers can produce different discretized variables if one, e.g., employs a supervised discretization technique like MDL-based discretization [31]. Similarly, different feature subsets can also be selected for different base classifiers using a supervised method such as the Correlation-Based-Feature subset selection algorithm (CFS) [32]. This enables the FMC to take a more

fine-grained class context into account when performing preprocessing as compared to, e.g., some MBCs approaches that rely on a single fixed set of feature variables. Furthermore, since preprocessing is only applied at the level of the base classifiers, standard supervised preprocessing techniques, such as the ones previously mentioned, can be directly employed (a property that is typically not available for other multi-label classifiers).

We have analyzed to what extent the flexibility of applying an individual preprocessing process to each base classifier is exploited by the learned classifiers. Table 1 shows the results for running supervised MDL discretization and CFS feature subset selection for the datasets analyzed in Section 5. For the CFS algorithm a best first search procedure has been used. The first row shows the total number of class pairs for each dataset, corresponding to the number of base classifiers to be learned as well as the number of different executions of the preprocessing methods. The second row shows the average number of different discretizations of the continuous variables produced by the discretization process for each base classifier; the average is taken wrt. the number of continuous variables. The third row in the table shows the number of different feature subsets selected for each base classifier on each dataset. From the table we see that FMC can indeed obtain different discretizations and feature subset selections when multiple runs of the preprocessing processes are performed for the different base classifiers. The extend to which this is achieved varies, though: comparatively minor variation is seen in, e.g., the number of different discretizations for the data sets *birds* and *emotions*, whereas larger variation is seen for the remaining data sets. Regarding FSS, with the exception of the *CAL500* dataset, we see that we almost always obtain a different feature subset for each base classifier.

	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
No. of class pairs	171	15051	36	15	1378	351	990	15	231	91
Avg. no. of disc	4.78	27.36	33.25	6.36	-	-	-	12.30	-	15.13
Avg. no. of FSS	116	1171	36	15	1264	344	818	15	231	88

Table 1: The first row of the table shows the number of class pairs in the full FMC classifier for the different data sets explored in Section 5. The second row in the table shows the average number of different discretizations of the continuous variables; the data sets *enron*, *genbase*, and medical do not contain any continuous variables. The third row in the table shows the number of different feature subsets selected for the different data sets. Note that the results in the table are independent of the particular choice of base classifier.

4. Scalable Learning and Inference

The complexity of learning and performing inference in the FMC is determined by the two stages of the algorithm:

- The computational complexity of doing learning and inference using all pairwise base classifiers is quadratic in the number of class variables. Concretely, if we consider all the class pairs, we have a total of $m(m-1)/2$ base classifiers. This phase of the algorithm is also influenced by the complexity of the selected base classifier, $T(n)$, characterized by the number of predictive attributes thus the final complexity of the first stage of an instantiated FMC is given by $\mathcal{O}(m^2) \cdot T(n)$. For instance, learning a naive Bayes classifier is linear in the number of predictive attributes and so the overall complexity of the first stage of the algorithm would be $\mathcal{O}(m^2 n)$. However, employing other base classifiers may increase the complexity of this phase of the algorithm.

- Performing exact inference in the induced MRF is NP-hard [33, 34], and with a complete model the computational complexity becomes exponential in the number of class variables making inference infeasible for domains with even a moderate number of class variables. The computational complexity is, however, independent of the number of feature variables. Since the number of feature variables is often far larger than the number of class variables (cf. Table 2), in practice we thus see that the computation time used by the two phases of the algorithm are comparable. This is also confirmed by the experimental results in Section 5.

One strategy for reducing the complexity of the first stage is to exploit that the overall learning setting is trivially parallelizable; we observe that most types of standard base classifiers can be learned independently, although one could envision complex classifiers that share substructures across class variable pairs. Thus, this part of the learning procedure is adaptable to most parallel computation architectures such as the MapReduce architecture [35]. For the second stage of the algorithm, one might consider a suitable approximate inference algorithm when exact inference is not feasible.

However, even when using a MapReduce architecture and approximate inference algorithms, the computational complexity of learning the base classifiers may still be demanding and similarly so for doing inference in the induced MRF. One immediate approach to overcome this difficulty is to only consider a restricted candidate subset of class variable pairs for which base classifiers should be learned. This strategy will not only reduce the complexity of the first stage of the framework, but will also reduce the complexity of the MRF model. In addition, it is possible that selecting a subset of class pairs may improve the quality of the results, as pruning edges from the MRF model may reduce the noise introduced by edges representing spurious class dependencies.

A straightforward strategy for selecting this candidate subset of class pairs could be to first measure the dependence/affinity between every class pair and then make a search-based selection. We propose two strategies that are based on measuring the empirical mutual information between all class variable pairs:

$$MI(C_i, C_j) = \sum_{C_i} \sum_{C_j} P(C_i, C_j) \log \left(\frac{P(C_i, C_j)}{P(C_i)P(C_j)} \right).$$

Our first proposal is to build a maximum spanning tree, using the Chow-Liu algorithm [36], over the class variables. This will be referred to as the *CLtree* algorithm in the remainder of the paper. This approach reduces the complexity of the algorithm to $\mathcal{O}(m) \cdot T(n)$, as it always selects a fixed-sized subset of $m-1$ base classifiers; in addition, the resulting MRF is guaranteed to be connected. Another strategy is to greedily select (according to the mutual information) the best subset of k class pairs with k being proportional to the number of classes m in the dataset. This latter strategy, referred to as *bestk*, would learn more complex models than the *CLtree* approach but still retain linear computational complexity $\mathcal{O}(km) \cdot T(n)$. However, in the resulting MRF model, we may end up with a disconnected class structure with some components containing single class variables; our model can deal with this by learning a single class classifier for each of these singleton variables and then set up the obtained distributions as the node potentials of the disconnected variables in the MRF.

The described approaches can potentially provide a drastic reduction in the complexity of the algorithm. However, given that the computed empirical mutual information employed in the selection strategies only capture marginal dependencies among a subset of the class pairs, we may unintentionally miss important dependencies, and thus fail to learn and include crucial base classifiers and the corresponding edges in the MRF. For that reason, we propose a third strategy, where we first compute the Markov blankets (MBs) for each class over the space of class variables and next include all the dependencies revealed by the learned MBs in the MRF model. We have chosen the HITON-MB [37] algorithm to learn the MBs of the classes from the data, as it is a well known algorithm that has been used successfully in related works [19], for example, to learn the class subgraph of an MBC.

The HITON-MB algorithm computes the Markov blanket for a given variable by evaluating several candidate subsets of variables to be included in the MB using a G^2 independence test. This algorithm must be parameterized by fixing a maximum number of elements to be tested in each possible subset of dependent variables; in this work we have selected a maximum subset size of 3 as this value has shown to provide good performance [37]. The procedure to select the candidate subset of class pairs using HITON-MB starts by running the algorithm for each class variable and learning its MB. Next, all class pairs appearing in the resulting MBs are included in the model. This strategy, which we will refer to as *hiton*, takes a different approach to alleviating the computational complexity as compared to the *CLtree* and the *bestk* strategies. Since *hiton* selects a variable number of pairs according to the dependencies found in the dataset and not according to a fixed-size set, the scalability gain will vary depending on the problem. The resulting models are therefore expected to obtain superior results at the expense of efficiency.

Lastly, we propose a different strategy for pruning the model, which is based on a forward greedy search algorithm guided by the BDeu scoring metric [38] for directed graphical models; a consequence of the search procedure is that we obtain a directed structure over the class variables, which we afterwards translate into an undirected structure by simply dropping the directions on the

arcs. Since the BDeu metric is decomposable [4], we can efficiently compute the score of a model as the sum of the local scores for the different nodes, which, in turn, only depends on the node in question and its parent set. For instance, the BDeu score with equivalent sample size α for a variable C_i with parent variables $Pa(C_i)$ is given by

$$BDeu(C_i \mid Pa(C_i) : \mathbf{D}) = \sum_{j=1}^{|Pa(C_i)|} \left(\log \frac{\Gamma(\alpha_{ij})}{\Gamma(N_{ij} + \alpha_{ij})} + \sum_{k=1}^{|C_i|} \log \frac{\Gamma(N_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})} \right), \quad (3)$$

where the first sum is over the joint states of the parent set $Pa(X_i)$ and the second one is over the states of the variable X_i; N_{ijk} corresponds to the number of records in $\mathbf{D}$ such that X_i is in its kth state and its parents in their jth configuration, and $N_{ij} = \sum_k N_{ijk}$. Similarly, α_{ij} is equal to α divided by the number of joint states over $Pa(X_i)$ and α_{ijk} is equal to α divided by the number of joint states of X_i and the parents in $Pa(X_i)$.

The proposed search procedure, referred to as *FSBDeu*, is outlined in Algorithm 1. The process can be separated into three different phases guided by heuristic strategies with the purpose of reducing the number of evaluations and to constrain the problem in order to enable a scalable search.

1. (Lines 1 to 3) First, the algorithm defines a topological ordering σ over the class variables by computing the score difference between the marginal BDeu score and the conditional BDeu score for each class pair; the position of a variable in the ordering is determined by the sum of the score differences computed for all class pairs in which the variable is involved. This value measures the overall dependence that each class variable is involved in. This step retains most of the computational burden of the procedure as it requires a quadratic number of evaluations given the number of labels.
2. (Lines 4 to 11) Afterwards, the candidate variable pairs are generated according to the ordering σ: For each pair of classes C_i and C_j the direction of the arc is established by the relative position of C_i and C_j in σ with the preceding variable becoming the head of the arc. This ordering guarantees that there will be no directed cycles in the resulting model even if all possible pairs are added.
3. (Lines 12 to 20) Finally a greedy forward search is performed: The pairs are sorted by their score and are tested once following the computed order. A pair (C_i, C_j) is introduced in the model if the score difference resulting from adding C_i to $Pa(C_j)$ is positive. When updating the parent sets, the corresponding BDeu scores are updated as well. The final structure is obtained by ignoring the directions of the arcs from the resulting model, notice that the parent sets are only considered to measure de BDeu score.

5. Experimental Evaluation

In the first part of this section we evaluate the scalability proposals described in the previous section in order to analyze the efficiency improvements

Algorithm 1: The algorithm implements a forward greedy search using the BDeu metric for selecting a reduced set of class variable pairs to be used in a FMC model.

Input: $\mathbf{D_C}$: Subset of the training data over classes $\mathbf{C} = \{C_1, \ldots, C_m\}$
Output: A collection $\mathbf{P}$ of selected class pairs variables (C_i, C_j)

`//Compute the score difference for each class pair.`
for each $C_i, C_j \in \mathbf{C}, j > i$ compute $diff_{C_i,C_j} = BDeu(C_j \mid C_i) - BDeu(C_j \mid \varnothing)$;

`//Establish a topological ordering among the classes according to the sum of all class pair scores in which the class is involved`
for each $C_i \in \mathbf{C}$ compute $sumDiff_{C_i} = \sum_{j=1, j\neq i}^{m} (diff_{C_i,C_j})$;
$\sigma \leftarrow$ order the variables $C_i \in \mathbf{C}$ by maximizing $sumDiff_{C_i}$;

`//Create the candidate pairs according to the topological ordering`
$\mathbf{P} \leftarrow \varnothing$;
for $C_i, C_j \in \mathbf{C}$, $j > i$ **do**
    if $\sigma_{C_i} > \sigma_{C_j}$ **then**
        Add (C_i, C_j) to $\mathbf{P}$
    else
        Add (C_j, C_i) to $\mathbf{P}$
    end
end

`//Sort the pairs according to the score differences`
$\mathbf{P} \leftarrow$ order each $(C_i, C_j) \in \mathbf{P}$ by maximizing $diff_{C_i,C_j}$;

`//Initialize empty parentsets for each class`
for $C_i \in \mathbf{C}$: $Pa(C_i) \leftarrow \varnothing$;

`//Perform a forward search over the ordered pairs`
for $(C_i, C_j) \in \mathbf{P}$ **do**
    if $\mathrm{BDeu}(C_j, \mid \mathrm{Pa}(C_j) \cup \{C_i\}) - \mathrm{BDeu}(C_j, \mid \mathrm{Pa}(C_j)) > 0$ **then**
        $Pa(C_j) \leftarrow Pa(C_j) \cup \{C_i\}$
    else
        Remove (C_i, C_j) from $\mathbf{P}$
    end
end
return P

and how the quality varies compared to the full pairwise framework. We also analyze which of the proposed strategies entails the most significant improvements regarding the trade-off between efficiency and quality. Secondly, we perform a comparison between our selected models and a collection of representative state-of-the art multi-label classifiers taken from the literature.

5.1. Experimental Set-up

We evaluate our approach using different instantiations of the proposed framework by taking two well-known Bayesian network classifiers as base classifiers: naive Bayes (NB), being a simple classifier with light computational and memory requirements, and A1DE [39], which is a more expressive classifier but also more computationally intensive.

We have built a prototype of the proposed framework that is based on two different platforms: The first step of the classifier has been implemented using the Mulan [40] library for multi-label dataset management as well as Weka [41] for learning the base classifiers. For the second step we use the UGM[6] Matlab package for performing inference over the induced MRF. A working pipeline of this process is publicly available in the previously mentioned companion website. The experiments were conducted on a dedicated Linux server with a Pentium Xeon 3.0 Ghz processor and 16GB of RAM.

As performance indicators we use the two metrics described in Section 2, global accuracy (*acc*) and Hamming accuracy (H_{acc}). By using both measures we can provide a clear overview of the behavior of each classifier as well as capture the difficulty of the problem; H_{acc} being an easier metric to optimize than *acc* especially for complex domains.

The experiments have been carried out using a collection of publicly available datasets, most of them taken from the Mulan repository[7]. We have selected datasets with a moderate number of labels and attributes, as it is infeasible to measure global accuracy for domains with a large number of labels. Additionally, we have included the *CLEF14* dataset in the experiments.[8] This dataset comes from a real world challenge in the field of computer vision [42], and contains seven binary classes as well as one multidimensional variable with 10 states, thus moving the problem from a multi-label domain to a multidimensional one. The characteristics of the datasets used in the experiments can be found in Table 2.

Since the standard version of the A1DE classifier cannot handle numeric variables we have preprocessed the datasets by discretizing the continuous variables.[9] This has been done by applying the procedure mentioned in Section 3.1, where we take advantage of the structure of the model and apply supervised discretization and posterior feature subset selection to each individual base

[6] `http://www.di.ens.fr/~mschmidt/Software/UGM.html`

[7] `http://mulan.sourceforge.net/datasets.html`

[8] `http://www.rovit.ua.es/dataset/vidrilo.html`

[9] We have conducted preliminary experiments using NB with Gaussian distributions as well as variants of the A1DE classifier capable of dealing with continuous variables [43]. However the accuracy results were not competitive with those achieved using discretization.

Database	Classes	Features	Instances
Birds	19	260	645
CAL500	174	68	502
Emotions	6	72	593
Enron	53	1001	1702
Genbase	27	1186	662
Medical	45	1449	978
Scene	6	294	2407
TMC2007	22	500	28596
Yeast	14	103	2417
CLEF14*	7x2/1x10	360	9500

Table 2: Datasets used in the evaluation. *The *CLEF14* dataset contains 7 binary classes and 1 class with 10 states.

classifier. In particular, we have discretized the numerical features using MDL-based discretization and applied the Correlation-Feature-Subset (CFS) selection algorithm using a best first search procedure.

5.2. Comparison Between Pruning Strategies

In Tables 3 and 4 we report on the results regarding *acc* and H_{acc}, respectively, for the full pairwise model and for each of the scalability strategies. The results have been obtained by performing a 10-fold cross validation for each of the datasets; the results highlighted in bold are the best for the corresponding datasets. At first sight, we can observe that there is a clear difference in the results obtained when using the different base classifiers, with A1DE obtaining superior results as expected. However, if we compare the approaches among themselves when using the same base classifier we can not observe a significant difference in quality between the full pairwise model and the pruned ones regarding both accuracy measures. To extend our comparison, we have performed statistical tests for both global accuracy and Hamming accuracy according to the procedure described in [44].

A Friedman test [45] comparing all the pruning approaches, using A1DE as base classifier, and with a 5% significance level, does not reject the hypothesis that all classifiers are equivalent with p-value = 0.1046 for global accuracy. However, in the case of Hamming accuracy the test is rejected with p-value = 0.0108. A post-hoc test with a 5% significance level using the Holm correction [46] shows that the *CLtree* approach obtains statistically worse results in case of Hamming accuracy when compared to the other approaches with p-value = 0.0095. Figure 2 shows the distribution of the mean ranks computed for the Friedman test for each of the approaches, confirming that the *CLtree* has the overall worst performance and that *FSBDeu*, *hiton*, and the full pairwise model have a comparable behavior.

Given the previous statistical tests, we conclude that most of the pruning approaches obtain results comparable to the full pairwise model, especially in the case of global accuracy. Next we will study and evaluate their results in

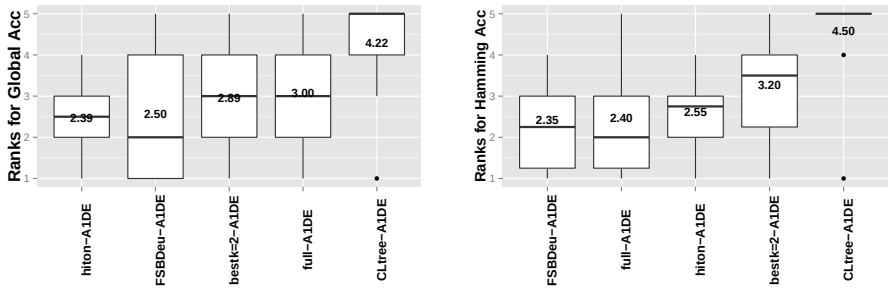

Figure 2: Boxplot representing the ranks obtained for each dataset for each of the methods evaluated for *acc* (left) and H_{acc} (right). The values correspond to the mean rank among each dataset as computed for the Friedman test.

terms of efficiency and general scalability. Figures 3 and 4 show the number of pairs selected for each method and dataset. In these plots we see that the full pairwise approach builds the largest models, and thus it is more inefficient in both phases of the algorithm, especially for datasets with a high number of classes. On the other hand, the Chow-Liu approach, *CLtree*, builds the least dense models, followed by *FSBDeu*, *bestk* with $k = 2m$, and *hiton*. In order to measure the efficiency of each pruning strategy we must not only look at the number of selected pairs but also at the cost of the overall method; in this case, *hiton* is by far the most costly strategy as the complexity of the HITON-MB algorithm itself is exponential in the number of class pairs. On the other hand, the *FSBDeu* approach is linear given the number of class pairs as the procedure is greedy and so the number of metric statistics to be computed is fixed. This is also the case for the *bestk* strategy and for the *CLtree* approach.

In conclusion, we propose the *FSBDeu* approach as the most promising strategy, as it combines both an efficient learning scheme and the ability to learn less dense models than both *hiton* and *bestk*. This strategy also has the important advantage of being parameter free unlike *hiton* and *bestk*. Lastly, the *CLtree* approach share computational benefits with *FSBDeu*, but if we also consider the accuracy results in Tables 3 and 4, we can observe that it generally obtains worse results; it can, however, be considered a suitable approach if efficiency is more important than accuracy. Finally, we would like to point to an important detail regarding the *FSBDeu* and the *CLtree* strategies for the *CLEF14* dataset wrt. *acc*. From Table 3 we see that the *CLtree* approach obtains the best results for this dataset, whereas the full pairwise model is the worst. This behavior reflects the underlying model for this dataset, which can be accurately described by a tree structure, resulting in the potential addition of noise by other methods. Observe that the *FSBDeu* approach is also capable of learning a suitable model for this dataset.

We performed an additional experiment to compare the complexity of both phases of the algorithm by measuring their contributions to the overall runtime. Figure 5 shows the runtime breakdown for the *full pairwise*, *FSBDeu* and *CLtree*

FMC	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-A1DE	0.4968	0.0000	0.3683	0.3017	0.1269	**0.9623**	0.6431	**0.6597**	0.2440	0.1850
bestk=2-A1DE	**0.5030**	0.0000	0.3937	0.3034	0.1257	0.9578	0.6647	0.6385	0.2450	0.1990
CLtree-A1DE	0.4781	0.0000	**0.4257**	0.2900	0.1263	0.9578	0.6360	0.5928	0.2336	0.1787
FSBDeu-A1DE	0.4877	0.0000	0.4134	**0.3253**	0.1263	0.9608	0.6339	0.6581	**0.2473**	**0.1994**
hiton-A1DE	0.4999	0.0000	0.3759	0.3034	**0.1275**	0.9593	**0.6666**	0.6581	0.2443	0.1928

FMC	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-NB	**0.4658**	0.0000	0.1383	0.2831	0.1105	0.9562	0.6227	0.5052	0.2320	0.1486
bestk=2-NB	0.4471	0.0000	0.1599	**0.2950**	0.1175	0.9562	0.6513	**0.5289**	0.2320	0.1631
CLtree-NB	0.4438	0.0000	**0.1896**	0.2731	0.1140	0.9547	0.6288	0.4811	0.2192	0.1399
FSBDeu-NB	0.4315	0.0000	0.1755	0.2933	**0.1181**	0.9562	0.6227	0.5069	**0.2336**	**0.1684**
hiton-NB	0.4625	0.0000	0.1407	0.2932	0.1169	**0.9608**	**0.6574**	0.5069	0.2331	0.1564

Table 3: Global accuracy of the FMC framework for the different pruning strategies and by using A1DE (above) and NB (below) as base classifier. Highlighted results show the best result for the corresponding dataset. Notice that owing to the large number of class variables in the CAL500 dataset, it is infeasible to provide accurate predictions of the full vector and thus the exact accuracy is very low.

FMC	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-A1DE	0.9521	0.8635	0.8545	0.7960	0.9511	**0.9986**	0.9892	**0.9122**	0.9329	**0.7939**
bestk=2-A1DE	**0.9523**	0.8626	0.8611	0.7965	0.9503	0.9984	0.9893	0.9067	0.9321	0.7938
CLtree-A1DE	0.9488	0.8629	**0.8668**	0.7892	0.9493	0.9983	0.9887	0.8985	0.9303	0.7872
FSBDeu-A1DE	0.9490	**0.8638**	0.8647	**0.7987**	0.9508	0.9985	0.9888	0.9117	**0.9329**	0.7937
hiton-A1DE	0.9514	0.8637	0.8566	0.7968	**0.9513**	0.9984	**0.9895**	0.9117	0.9328	0.7931

FMC	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-NB	**0.9449**	**0.8619**	0.7621	0.7835	0.9309	**0.9984**	0.9888	0.8853	**0.9302**	0.7770
bestk=2-NB	0.9388	0.8607	0.7715	**0.7852**	0.9379	0.9983	0.9890	0.8839	0.9292	**0.7796**
CLtree-NB	0.9363	0.8599	**0.7858**	0.7822	0.9370	0.9982	0.9884	0.8718	0.9271	0.7680
FSBDeu-NB	0.9347	0.8616	0.7798	0.7846	0.9361	0.9983	0.9884	**0.8858**	0.9299	0.7792
hiton-NB	0.9388	0.8617	0.7643	0.7815	**0.9417**	**0.9984**	**0.9892**	**0.8858**	**0.9302**	0.7752

Table 4: Hamming accuracy of the FMC framework for the different pruning strategy. Highlighted results show the best result for the corresponding dataset.

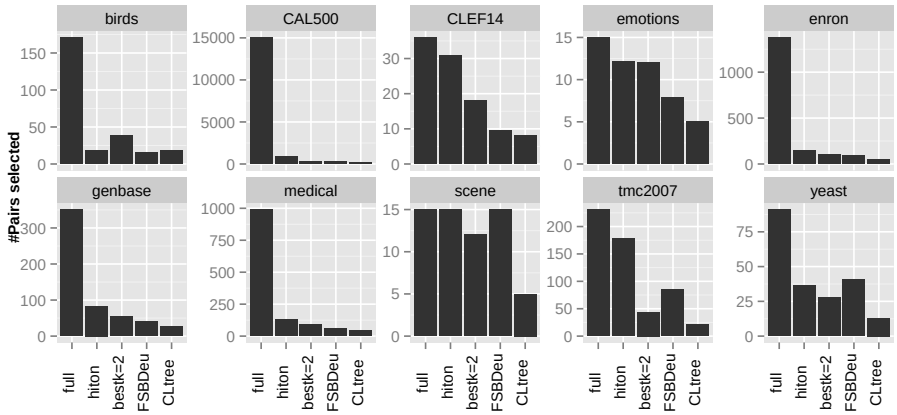

Figure 3: Number of pairs selected by each pruning strategy and for each dataset.

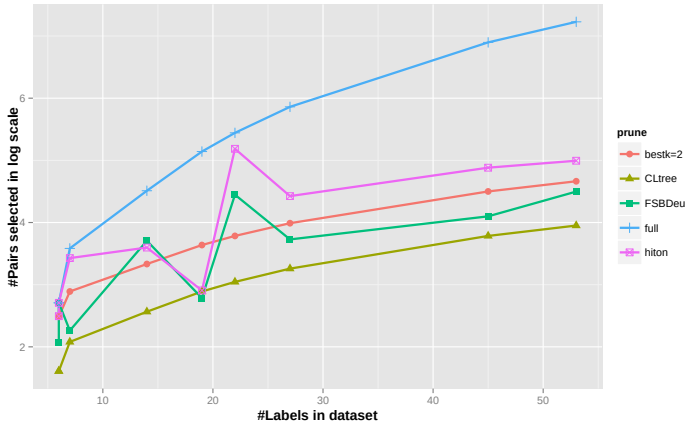

Figure 4: Number of selected pairs in log scale (y-axis) according to the number of labels in the datasets (x-axis) for each pruning strategy. This figure shows how the FMC framework scales up when different pruning approaches are used as the number of labels in the dataset grows. The comparison show that the quadratic complexity of the full model dominates the pruning-based approaches.

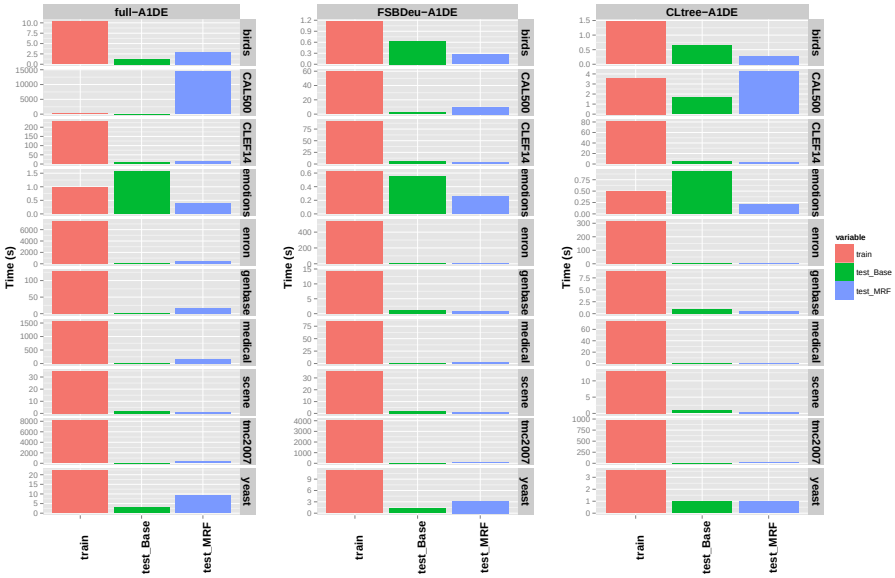

Figure 5: Breakdown of the runtime in seconds for the different pruning strategies and datasets. Each plot shows the training time of the first stage, followed by the test time, still from the first stage, and the test time from the second stage. All the results are obtained as the average from 10-fold cross validation using A1DE as the base classifier. Notice that the time scale is different for each dataset.

approaches for all the datasets: The first column corresponds to the training time for the first stage of the algorithm; the second column shows the time for classifying the test instances using all base classifiers; finally the third column corresponds to the time used by the second stage, where inference is performed in the learned MRF for each test instance (see also the discussion below). All results are expressed in seconds and have been obtained by averaging the runtime based on 10-fold cross validation and using A1DE as base classifier. As we can observe, in practice the first stage is less efficient than the second one, especially when a pruning strategy is being used. There are only two exceptions to this observation: *emotions*, which is a very small dataset and the training time is therefore correspondingly low, and *CAL500*, which features a large number of labels and instances, thus complicating the MRF inference procedure. It should be noted that the tests are based on the A1DE classifier, and the observed differences would therefore be reduced by using the NB instead.

A final point to consider regarding efficiency is the MRF inference approach employed to obtain the final predictions. The reported results have been obtained using approximate inference. Specifically, we have used the loopy belief propagation algorithm [47], which is included in the above mentioned Matlab package. Performing approximate inference over a MRF introduces a significant improvement in the computational requirements of the algorithm at the expense of a potentially detriment to the quality of the predictions. For that

Dataset	full-A1DE		FSBDeu		CLtree	
	LBP	exact	LBP	exact	LBP	exact
birds	0.9521	0.9521	0.9490	0.9490	0.9488	0.9488
CLEF14	0.8548	0.8543	0.8647	0.8579	0.8668	0.8520
emotions	0.7960	0.7960	0.7987	0.7987	0.7892	0.7892
enron	0.9511	-	0.9508	-	0.9493	0.9493
genbase	0.9986	0.9982	0.9985	0.9985	0.9983	0.9983
medical	0.9892	-	0.9888	0.9909	0.9887	-
scene	0.9117	0.9114	0.9117	0.9114	0.8985	0.8985
tmc2007	0.9329	0.9323	0.9329	0.9317	0.9303	0.9303
yeast	0.7939	0.7938	0.7937	0.7937	0.7872	0.7872

Dataset	full-A1DE		FSBDeu		CLtree	
	LBP	exact	LBP	exact	LBP	exact
birds	0.4968	0.4968	0.4877	0.4877	0.4781	0.4781
CLEF14	0.3701	0.3699	0.4134	0.3646	0.4257	0.3483
emotions	0.3017	0.3017	0.3253	0.3253	0.2900	0.2900
enron	0.1269	-	0.1263	-	0.1263	0.1263
genbase	0.9623	0.9501	0.9608	0.9608	0.9578	0.9578
medical	0.6431	-	0.6339	0.6735	0.6360	0.6360
scene	0.6581	0.6568	0.6581	0.6568	0.5928	0.5928
tmc2007	0.2440	0.2390	0.2473	0.2430	0.2336	0.2336
yeast	0.1850	0.1850	0.1994	0.1994	0.1787	0.1787

Table 5: Hamming accuracy (above) and global accuracy (below) for the selected models and datasets when using exact and approximate inference.

reason we have conducted additional experiments running exact inference for some of the smaller datasets and pruned models. Table 5 shows the accuracy results obtained for the full pairwise model and the *FSBDeu* and *CLtree* approaches using the A1DE base classifier. From the results we see that there is no clear difference when it comes to the final predictions using either exact or approximate inference. However, when it comes to the complexity of the procedures the differences are significant. Figure 6 shows the runtime results for the previous experiment, revealing that, as expected, the approximate inference algorithm is many times faster than the exact approach.

5.3. Comparison with Other State-of-the Art classifiers

In order to evaluate the performance of our proposal we have replicated our experimental evaluation for three well known state-of-the art multilabel classifiers, all of which have public implementations available: Binary-Relevance (BR), Ensembles of Classifier Chains (ECC), and RAkEL [11]. The three classifiers are also instantiated with a particular base classifier, and here we have selected the same base classifiers as for our original proposal (naive Bayes and A1DE). Regarding the parameterization of these approaches, ECC has been configured to learn 10 different models for the ensemble, and for RAkEL we have used the recommended configuration [11] with $2m$ models having triplets of label combinations $k = 3$. We have used the open implementation from the Mulan library for the experiments.

Tables 6 and 7 show the results for global accuracy and Hamming accuracy, respectively, for each of the classifiers instantiated with both base classifiers;

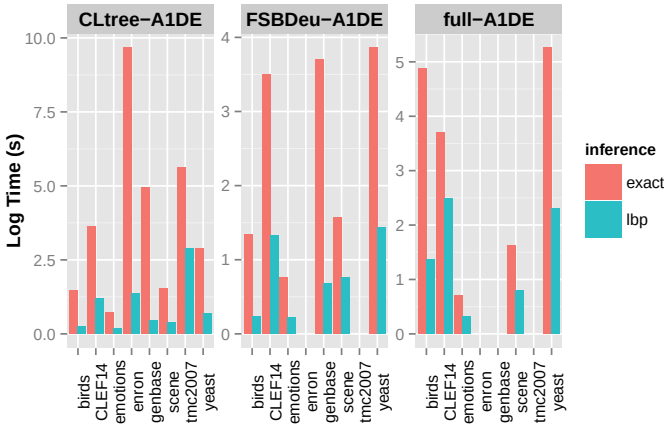

Figure 6: Logarithmic runtime measured in seconds for the selected models and datasets when using exact and approximate inference.

we have included our full pairwise *FMC* approach and the BDeu pruned model *FMC-BDeu* for the sake of comparison. We have also included the same preprocessing scheme described for our framework; the three straw-men methods are also transformation-based methods that learn independent base classifiers for which we can apply separate single class MDL discretization and CFS feature selection. However, we have observed that, in some domains, using FSS decreases the performance of the multilabel classifiers when using A1DE as base classifier, especially in terms of global accuracy. For the *emotions* and *yeast* datasets we have therefore replaced the results in Tables 6 and 7 with the corresponding results obtained without using FSS. This was, however, not feasible for the data sets *enron, medical*, and *genbase*, as the algorithms are not able to handle these datasets using the original full set of features.

This behavior can be explained by the internal workings of these multi-label classifiers. In the case of transformation methods such as Binary Relevance and RAkEL, the final output is computed by relying only on the individual class predictions of the base classifiers. Thus, any change in accuracy of these models, either positive or negative, can result in significant variations in the final multilabel prediction. On the other hand, our FMC classifier uses the computed probability distributions from the probabilistic base classifiers instead of the individual predictions, so it is less sensitive to variations of the overall accuracy of its base classifiers.

In general, we can observe that our proposed method obtains superior results when compared to the other state-of-the art classifiers. To extend the comparison we have performed statistical tests following the same procedure as employed in the previous subsection. The Friedman test with a 5% significance level rejects the hypothesis that all classifiers are equivalent with a $p-$value of $2.6848e^{-05}$ in the case of global accuracy and with a $p-$value of $1.0862e^{-05}$ for

Classifier	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-A1DE	**0.4968**	**0.0000**	0.3683	0.3017	**0.1269**	0.9623	**0.6431**	**0.6597**	0.2440	0.1850
full-NB	0.4658	**0.0000**	0.1383	0.2831	0.1105	0.9562	0.6227	0.5052	0.2320	0.1486
FSBDeu-A1DE	0.4877	**0.0000**	**0.4134**	**0.3253**	0.1263	0.9608	0.6339	0.6581	**0.2473**	**0.1994**
FSBDeu-NB	0.4315	**0.0000**	0.1755	0.2933	0.1181	0.9562	0.6227	0.5069	0.2336	0.1684
BR-A1DE	0.4564	**0.0000**	0.2286	*0.2377	0.0000	0.9155	0.1124	*0.5451	0.1350	*0.1117
BR-NB	0.4348	**0.0000**	0.0814	0.2495	0.1093	**0.9758**	0.5797	0.3079	0.2113	0.1039
ECC-A1DE	0.4566	**0.0000**	0.1463	*0.3019	0.0000	0.9185	0.1154	*0.5588	0.1478	*0.1572
ECC-NB	0.4562	**0.0000**	0.1033	0.3001	0.1234	0.9728	0.6205	0.3403	0.2206	0.1643
RAKEL-A1DE	0.0312	**0.0000**	0.3746	*0.3236	0.0000	0.0000	0.0143	*0.5376	0.1412	*0.1270
RAKEL-NB	0.4564	**0.0000**	0.1221	0.2983	0.1051	0.9698	0.6032	0.4674	0.2127	0.1275

Table 6: Global accuracy for the literature classifiers and the FMC. Highlighted results show the best results for the corresponding dataset. Results marked with a '*' has been obtained without using FSS.

Classifier	birds	CAL500	CLEF14	emotions	enron	genbase	medical	scene	tmc2007	yeast
full-A1DE	**0.9521**	0.8635	0.8545	0.7960	**0.9511**	0.9986	**0.9892**	**0.9122**	0.9329	**0.7939**
full-NB	0.9449	0.8619	0.7621	0.7835	0.9309	0.9984	0.9888	0.8853	0.9302	0.7770
FSBDeu-A1DE	0.9490	**0.8638**	0.8647	0.7987	0.9508	0.9985	0.9888	0.9117	**0.9329**	0.7937
FSBDeu-NB	0.9347	0.8616	0.7798	0.7846	0.9361	0.9983	0.9884	0.8858	0.9299	0.7792
BR-A1DE	0.9472	0.8503	0.8364	0.7797	*0.9364	0.9965	0.9755	*0.9024	0.9148	*0.7593
BR-NB	0.9322	0.8545	0.7245	0.7796	0.9312	**0.9990**	0.9878	0.8470	0.9267	0.7572
ECC-A1DE	0.9478	0.8503	0.8047	0.7897	*0.9366	0.9968	0.9756	*0.9048	0.9167	*0.7727
ECC-NB	0.9442	0.8526	0.7432	0.7909	0.9324	0.9989	0.9888	0.8548	0.9267	0.7812
RAKEL-A1DE	0.8427	0.6520	**0.8827**	***0.8027**	0.7308	0.7656	*0.9361	0.8965	0.9157	*0.7691
RAKEL-NB	0.9403	0.8620	0.7606	0.7908	0.9420	0.9989	0.9884	0.8809	0.9276	0.7684

Table 7: Hamming accuracy for the straw-men classifiers and the FMC. Highlighted results show the best result for the corresponding dataset. Results marked with a '*' correspond has been obtained without using FSS.

Hamming accuracy. Following this result we performed a post-hoc test using Holm's procedure with a 5% significance level. The results can be found in Table 8 together with the ranking computed for the Friedman test; this test compares all methods with the approach having the highest mean rank (FMC-full-A1DE for *acc* and FMC-FSBDeu-A1DE for H_{acc}) as control.

This post-hoc test reveals that both the full and pruned FMC classifier instantiated with A1DE are the best classifiers in general, obtaining a significant difference in the mean rank value. The classifiers are equivalent among themselves and statistically superior to the others for both *acc* and H_{acc}.

Finally we have evaluated the scalability properties of each approach by measuring their runtime for the configured experiment. Figures 7 and 8 show the averaged runtime based on cross validation for the training and testing phases, respectively. This comparison confirms that the full unpruned FMC model is by far the most complex and inefficient model. However, when including the FSBDeu strategy the runtime becomes equivalent to the other referenced methods in almost all the evaluated domains.

Global Accuracy (acc)		
Algorithm	Rank	p-value
FSBDeu-A1DE	1.89	-
full-A1DE	2.11	$\mathbf{8.7627e^{-01}}$
FSBDeu-NB	5.17	$4.3286e^{-02}$
ECC-NB	5.44	$3.8194e^{-02}$
full-NB	5.61	$3.6432e^{-02}$
ECC-A1DE	6.00	$1.9856e^{-02}$
RAKEL-NB	6.28	$1.2627e^{-02}$
RAKEL-A1DE	7.22	$1.3047e^{-03}$
BR-NB	7.56	$5.7415e^{-04}$
BR-A1DE	7.72	$3.9309e^{-04}$

Hamming Accuracy (H_{acc})		
Algorithm	Rank	p-value
full-A1DE	1.90	-
FSBDeu-A1DE	2.10	$\mathbf{8.8257e^{-01}}$
RAKEL-NB	5.60	$1.9862e^{-02}$
full-NB	5.70	$1.9862e^{-02}$
ECC-NB	5.80	$1.9862e^{-02}$
FSBDeu-NB	5.80	$1.9862e^{-02}$
ECC-A1DE	5.90	$1.8809e^{-02}$
BR-A1DE	7.20	$6.3464e^{-04}$
RAKEL-A1DE	7.30	$5.3269e^{-04}$
BR-NB	7.70	$1.6552e^{-04}$

Table 8: Results from the post-hoc test using Holm's procedure for both global accuracy and Hamming accuracy based on selected results from Tables 6 and 7. The table shows the ranking computed for the Friedman test and the adjusted p-value using Holm's procedure with a 5% significance level. Boldfaced results correspond to rejected hypotheses.

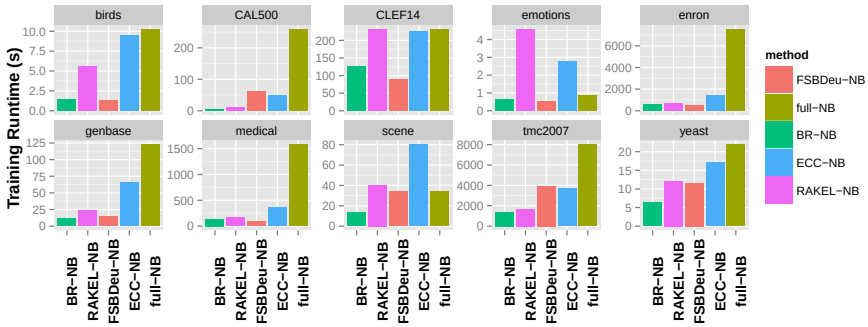

Figure 7: Mean training runtime in seconds for each method and dataset.

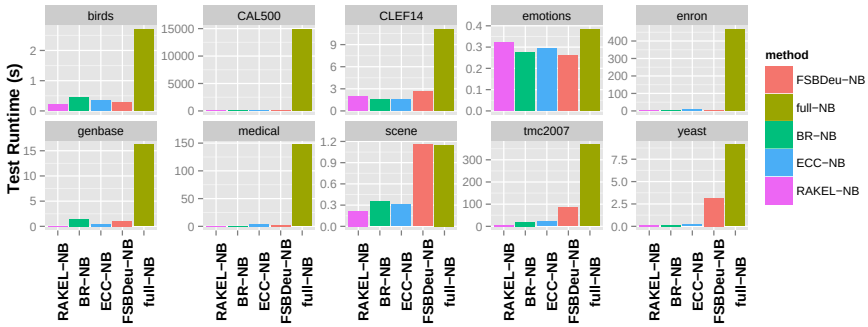

Figure 8: Mean test runtime in seconds for each method and dataset.

6. Conclusions

We have introduced a new framework for multidimensional classification based on the construction of a collection of pairwise classifiers that in combination induce a Markov random field in which multi-label inference can be performed. In our experiments, we have compared our proposal with a range of available state-of-the art classifiers and obtained favourable results. A potential concern about the framework is its scalability. We have outlined several strategies to address this problem based on pruning class variable pairs or the induced MRF model. The experiments show that the complexity of our approach can be drastically reduced while maintaining competitive results.

In future work, we will study the dependency relationship between classes and extend our experiments to additional multilabel and multidimensional domains with the purpose of further exploring the capabilities of this framework. We will look for conditional dependences between the class variables and learning specific base classifiers to maximize their impact in the overall framework performance. Finally, a generalization of the framework using ternary or n-dependences relations among the classes will also be considered.

Acknowledgments.

This work has been partially funded by FEDER funds and the Spanish Government (MINECO) through projects TIN2010-20900-C04-03 and TIN2013-46638-C3-3-P and by JCCM through project PEII-2014-049-P. Jacinto Arias is also funded by the MECD grant FPU13/00202.

References

[1] J. Arias, J. Gámez, T. D. Nielsen, J. Puerta, A pairwise class interaction framework for multilabel classification, in: L. van der Gaag, A. Feelders (Eds.), Probabilistic Graphical Models, Vol. 8754 of Lecture Notes in Computer Science, Springer International Publishing, 2014, pp. 17–32.

[2] J. Read, C. Bielza, P. L. naga, Multi-Dimensional Classification with Super-Classes, IEEE Transactions on knowledge and data engineering 26 (7) (2014) 1720–1733.

[3] M.-L. Zhang, Z.-H. Zhou, A review on multi-label learning algorithms, Knowledge and Data Engineering, IEEE Transactions on 26 (8) (2014) 1819–1837.

[4] F. V. Jensen, T. D. Nielsen, BAyesian Networks and Decision Graphs, 2nd Edition, Springer Publishing Company, Incorporated, 2007.

[5] C. Bielza, G. Li, P. Larrañaga, Multi-dimensional classification with Bayesian networks, International Journal of Approximate Reasoning 52 (6) (2011) 705–727. doi:10.1016/j.ijar.2011.01.007.

[6] Y. Guo, S. Gu, Multi-label classification using conditional dependency networks, in: International Joint Conference on Artificial Intelligence, Vol. 22 of IJCAI'11, AAAI Press, 2011, pp. 1300–1305.

[7] A. L. Madsen, F. Jensen, A. Salmerón, M. Karlsen, H. Langseth, T. D. Nielsen, A new method for vertical parallelisation of TAN learning based on balanced incomplete block designs, in: Probabilistic Graphical Models, Springer, 2014, pp. 302–317.

[8] E. Hüllermeier, J. Fürnkranz, W. Cheng, K. Brinker, Label ranking by learning pairwise preferences, Artificial Intelligence 172 (16-17) (2008) 1897–1916.

[9] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Džeroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognition 45 (9) (2012) 3084–3104. doi:10.1016/j.patcog.2012.03.004.

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

[11] G. Tsoumakas, I. Katakis, I. P. Vlahavas, Random k-labelsets for multilabel classification, IEEE Transactions on Knowledge and Data Engineering 23 (7) (2011) 1079–1089.

[12] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning 85 (3) (2011) 333–359.

[13] E. Alvares-Cherman, J. Metz, M. C. Monard, Incorporating label dependency into the binary relevance framework for multi-label classification, Expert Systems with Applications 39 (2) (2012) 1647–1655. doi:10.1016/j.eswa.2011.06.056.

[14] E. Montañes, R. Senge, J. Barranquero, J. Ramón Quevedo, J. José Del Coz, E. Hüllermeier, Dependent binary relevance models for multi-label classification, Pattern Recognition 47 (3) (2014) 1494–1508. doi:10.1016/j.patcog.2013.09.029.

[15] K. Dembczyński, W. Cheng, E. Hüllermeier, Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains, in: Proceedings of the 27th international conference on machine learning (ICML-10), 2010, pp. 279–286.

[16] L. C. van der Gaag, P. R. de Waal, Multi-dimensional Bayesian network classifiers, in: 3rd European Workshop on Probabilistic Graphical Models (PGM-06), 2006, pp. 107–114.

[17] P. R. de Waal, L. C. van der Gaag, Inference and learning in multi-dimensional Bayesian network classifiers, in: K. Mellouli (Ed.), Symbolic and Quantitative Approaches to Reasoning with Uncertainty, Vol. 4724 of LNCS, Springer Berlin Heidelberg, 2007, pp. 501–511.

[18] J. D. Rodríguez, J. A. Lozano, Multi-objective learning of multi-dimensional Bayesian classifiers, in: 8th International Conference on Hybrid Intelligent Systems (HIS 2008), 2008, pp. 501–506.

[19] H. Borchani, C. Bielza, P. Martínez-Martín, P. Larrañaga, Markov blanket-based approach for learning multi-dimensional Bayesian network classifiers: An application to predict the european quality of life-5 dimensions (EQ-5D) from the 39-item Parkinson's disease questionnaire (PDQ-39), Journal of Biomedical Informatics 45 (6) (2012) 1175–1184.

[20] H. Borchani, C. Bielza, C. Toro, P. Larrañaga, Predicting human immunodeficiency virus inhibitors using multi-dimensional Bayesian network classifiers, Artificial Intelligence in Medicine 57 (3) (2013) 219–229.

[21] J. C. Zaragoza, L. E. Sucar, E. F. Morales, A two-step method to learn multidimensional Bayesian network classifiers based on mutual information measures, in: 24th International Florida Artificial Intelligence Research Society Conference (FLAIRS-11), 2011.

[22] G. Corani, A. Antonucci, D. Mauá, S. Gabaglio, Trading off speed and accuracy in multilabel classification, in: Probabilistic Graphical Models, Vol. 8754 of Lecture Notes in Computer Science, Springer, 2014, pp. 145–159.

[23] A. Antonucci, G. Corani, D. Mauá, S. Gabaglio, An ensemble of Bayesian networks for multilabel classification, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI'13, AAAI Press, 2013, pp. 1220–1225.

[24] C. Hong, I. Batal, M. Hauskrecht, A Mixtures-of-Trees Framework for Multi-Label Classification, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management - CIKM '14, ACM Press, 2014, pp. 211–220. doi:10.1145/2661829.2661989.

[25] J. H. Zaragoza, L. E. Sucar, E. F. Morales, C. Bielza, P. Larrañaga, Bayesian chain classifiers for multidimensional classification, in: 22nd International Joint Conference on Artificial Intelligence (IJCAI'11), AAAI Press, 2011, pp. 2192–2197.

[26] L. E. Sucar, C. Bielza, E. F. Morales, P. Hernandez-Leal, J. H. Zaragoza, P. Larrañaga, Multi-label classification with Bayesian network-based chain classifiers, Pattern Recognition Letters 41 (2014) 14–22.

[27] G. Varando, C. Bielza, P. Larrañaga, Expressive power of binary relevance and chain classifiers based on Bayesian networks for multi-label classification, in: Probabilistic Graphical Models, Vol. 8754 of Lecture Notes in Computer Science, Springer, 2014, pp. 519–534.

[28] M. Ramírez-Corona, L. Sucar, E. Morales, Multi-label classification for tree and directed acyclic graphs hierarchies, in: Probabilistic Graphical Models, Vol. 8754 of Lecture Notes in Computer Science, Springer, 2014, pp. 409–425.

[29] M.-L. Zhang, K. Zhang, Multi-label learning by exploiting label dependency, in: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, 2010, pp. 999–1007. doi:10.1145/1835804.1835930.

[30] S. Wang, J. Wang, Z. Wang, Q. Ji, Enhancing multi-label classification by modeling dependencies among labels, Pattern Recognition 47 (10) (2014) 3405–3413. doi:10.1016/j.patcog.2014.04.009.

[31] U. M. Fayyad, K. B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93), 1993, pp. 1022–1029.

[32] M. A. Hall, Correlation-based feature selection for machine learning, Ph.D. thesis, The University of Waikato (1999).

[33] P. Dagum, M. Luby, Approximating probabilistic inference in Bayesian belief networks is np-hard, Artificial intelligence 60 (1) (1993) 141–153.

[34] P. Smyth, Belief networks, hidden Markov models, and Markov random fields: A unifying view, Pattern recognition letters 18 (11) (1997) 1261–1268.

[35] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, in: Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation - Volume 6, USENIX Association, 2004, pp. 137–150.

[36] C. K. Chow, C. Liu, Approximating discrete probability distributions with dependence trees, IEEE Transactions on Information Theory 14 (1968) 462–467.

[37] C. F. Aliferis, I. Tsamardinos, A. Statnikov, Hiton: a novel Markov blanket algorithm for optimal variable selection, in: AMIA Annual Symposium Proceedings, Vol. 2003, American Medical Informatics Association, 2003, p. 21.

[38] W. Buntine, Theory refinement on Bayesian networks, in: Proceedings of the Seventh conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1991, pp. 52–60.

[39] G. I. Webb, J. R. Boughton, Z. Wang, Not so naive Bayes: aggregating one-dependence estimators, Machine Learning 58 (1) (2005) 5–24.

[40] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, in: Data mining and knowledge discovery handbook, Springer, 2010, pp. 667–685.

[41] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA data mining software: an update, ACM SIGKDD Explorations Newsletter 11 (1) (2009) 10–18.

[42] J. Martinez-Gomez, M. Cazorla, I. Garcia-Varea, V. Morell, Overview of the ImageCLEF 2014 robot vision task, in: CLEF 2014 Evaluation Labs and Workshop, Online Working Notes, 2014.

[43] M. J. Flores, J. A. Gámez, A. M. Martínez, J. M. Puerta, GAODE and HAODE: two proposals based on AODE to deal with continuous variables, in: Proceedings of the 26th annual international conference on machine learning, ACM, 2009, pp. 313–320.

[44] J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal of Machine Learning Research 7 (2006) 1–30.

[45] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, The Annals of Mathematical Statistics (1940) 86–92.

[46] S. Holm, A simple sequentially rejective multiple test procedure, Scandinavian Journal of Statistics (1979) 65–70.

[47] K. P. Murphy, Y. Weiss, M. I. Jordan, Loopy belief propagation for approximate inference: An empirical study, in: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99), Morgan Kaufmann, 1999, pp. 467–475.