An Ensemble of Bayesian Networks for Multilabel Classification*

Antonucci Alessandro⋆, Giorgio Corani⋆, Denis Mauá⋆, and Sandra Gabaglio†
⋆Istituto "Dalle Molle" di Studi sull'Intelligenza Artificiale (IDSIA), Manno (Switzerland)
{alessandro,giorgio,denis}@idsia.ch
†Istituto Sistemi Informativi e Networking (ISIN), Manno (Switzerland)
sandra.gabaglio@supsi.ch

Abstract

We present a novel approach for multilabel classification based on an ensemble of Bayesian networks. The class variables are connected by a tree; each model of the ensemble uses a different class as root of the tree. We assume the features to be conditionally independent given the classes, thus generalizing the naive Bayes assumption to the multiclass case. This assumption allows us to optimally identify the correlations between classes and features; such correlations are moreover shared across all models of the ensemble. Inferences are drawn from the ensemble via logarithmic opinion pooling. To minimize Hamming loss, we compute the marginal probability of the classes by running standard inference on each Bayesian network in the ensemble, and then pooling the inferences. To instead minimize the subset 0/1 loss, we pool the joint distributions of each model and cast the problem as a MAP inference in the corresponding graphical model. Experiments show that the approach is competitive with state-of-the-art methods for multilabel classification.

1 Introduction

In traditional classification each instance belongs only to a single class. *Multilabel classification* generalizes this idea by allowing each instance to belong to different *relevant classes* at the same time. A simple approach to deal with multilabel classification is *binary relevance* (BR): a binary classifier is independently trained for each class and then predicts whether such class is relevant or not for each instance. BR ignores dependencies among the different classes; yet it can be quite competitive [Dembczynski *et al.*, 2012] especially under loss functions which decompose label-wise, such as the Hamming loss.

Instead the global accuracy (also called *exact match*) does *not* decompose label-wise; a classification is regarded as accurate only if the relevance of *every* class has been correctly identified. To obtain good performance under global accuracy, it is necessary identifying the *maximum a posteriori* (MAP) configuration, that is, the mode of the joint probability distribution of the classes given the features, which in turn requires to properly model the stochastic dependencies among the different classes. A state-of-the-art approach to this end is the classifier chain (CC) [Read *et al.*, 2011]. Although CC has not been designed to minimize a specific loss function, it has been recently pointed out [Dembczynski *et al.*, 2010; 2012] that it can be interpreted as a greedy approach for identifying the mode of the joint distribution of the classes given the value of the features; this might explain its good performance under global accuracy.

Bayesian networks (BNs) are a powerful tool to learn and perform inference on the joint distribution of several variables; yet, they are not commonly used in multilabel classification. They pose two main problems when dealing with multilabel classification: learning from data the structure of the BN model and performing inference on the learned model in order to issue a classification. In [Bielza *et al.*, 2011], the structural learning problem is addressed by partitioning the arcs of the structure into three sets: links among the class variables (*class graph*), links among features (*features graph*) and links between class and features variables (*bridge graph*). Each subgraph is separately learned and can have different topology (tree, polytree, etc). Inference is performed either by an optimized enumerative scheme or by a greedy search, depending on the number of classes.

Thoroughly searching for the best structure introduces the issue of model uncertainty: several structures, among the many analyzed, might achieve similar scores and thus the model selection can become too sensitive on the specificities of the training data, eventually increasing the variance component of the error. In traditional classification, this problem has been successfully addressed by instantiating all the BN classifiers whose structure satisfy some constraints and then combining their inferences, avoiding thus an exhaustive search for the best structure [Webb *et al.*, 2005]. The resulting algorithm, called AODE, is indeed known to be a high-performance classifier.

In this paper, we extend to the multilabel case the idea of averaging over a constrained family of classifiers. We assume the graph connecting the classes to be a tree. Rather than searching for the optimal tree, we instantiate a differ-

*Work supported by the Swiss NSF grant no. 200020_134759 / 1 and by the Hasler foundation grant n. 10030.

ent classifier for each class; each classifier adopts a different class as root. We moreover assume the independence of the features given the classes, thus generalizing to the multilabel case the naive Bayes assumption. This implies that the subgraph connecting the features is empty. Thanks to the naive-multilabel assumption, we can identify the *optimal* subgraph linking classes and the features in polynomial time [de Campos and Ji, 2011]. This optimal subgraph is the same for all the models of the ensemble, which speeds up inference and learning. Summing up: we perform no search for the class graph, as we take the ensemble approach; we perform no search for the feature graph; we perform an optimal search for the bridge graph.

Eventually, we combine the inferences produced by the members of the ensemble via geometric average (logarithmic opinion pool), which minimizes the average KL-divergence between the true distribution and the elements of an ensemble of probabilistic models [Heskes, 1997]. It is moreover well known that simple combination schemes (such as arithmetic average or geometric average) often dominates more refined combination schemes [Yang *et al.*, 2007; Timmermann, 2006].

The inferences are tailored to the loss function being used. To minimize Hamming loss, we query each classifier of the ensemble about the posterior marginal distribution of each class variable, and then pool such marginals to obtain an ensemble probability of each class being relevant. To instead maximize the global accuracy, we compute the MAP inference in the pooled model. Although both the marginal and joint inferences are NP-hard in general, they can often be performed quite efficiently by state-of-the-art algorithms.

2 Probabilistic Multilabel Classification

We denote the array of class events as $\mathbf{C} = (C_1, \ldots, C_n)$; this is an array of Boolean variables which expresses whether each class label is relevant or not for the given instance. Thus, $\mathbf{C}$ takes its values in $\{0,1\}^n$. A *instantiation* of class events is denoted as $\mathbf{c} = (c_1, \ldots, c_n)$. We denote the set of features as $\mathbf{F} = (F_1, \ldots, F_m)$. An instantiation of the features is denoted as $\mathbf{f} = (f_1, \ldots, f_m)$. A set of complete training instances $\mathcal{D} = \{(\mathbf{c}, \mathbf{f})\}$ is supposed to be available.

For a generic instance, we denote as $\hat{\mathbf{c}}$ and $\tilde{\mathbf{c}}$ respectively the labels assigned by the classifier and the actual labels; similarly, with reference to class c_i, we denote as $\hat{c}_i$ and $\tilde{c}_i^*$ its predicted and actual labels. The most common metrics to evaluate multilabel classifiers are the *global accuracy* (a.k.a. *exact match*), denoted as *acc*, and the Hamming loss (HL). Rather than HL, we use in the following the *Hamming accuracy*, defined as H_{acc}=1-HL. This simplifies the interpretation of the results as both *acc* and H_{acc} are better when they are higher. The metrics are defined as follows:

$$\mathrm{acc} = \delta(\hat{\mathbf{c}}, \tilde{\mathbf{c}}), \qquad \mathrm{H}_{acc} = n^{-1} \sum_{i=1}^{n} \delta(\hat{c}_i, \tilde{c}_i).$$

A probabilistic multilabel classifier classifies data according to a conditional joint probability distribution $P(\mathbf{C}|\mathbf{F})$. Our approach consists in learning an ensemble of probabilistic classifiers that jointly approximate the (unknown) true distribution $P(\mathbf{C}|\mathbf{F})$. When classifying an instance with features $\mathbf{f}$, two different kinds of inference are performed depending on whether the goal is maximizing global accuracy or Hamming accuracy. The most probable joint class $\hat{\mathbf{c}}$ maximizes global accuracy (under the assumption that $P(\mathbf{C}|\mathbf{F})$ is the true distribution), and is given by

$$\hat{\mathbf{c}} = \arg \max_{\mathbf{c} \in \{0,1\}^n} P(\mathbf{c}|\mathbf{f}). \tag{1}$$

The maximizer of Hamming accuracy instead consists in selecting the most probable marginal class label separately for each $C_i \in \mathbf{C}$:

$$\hat{c}_i = \arg \max_{c_i \in \{0,1\}} P(c_i|\mathbf{f}), \tag{2}$$

with $P(c_i|\mathbf{f}) = \sum_{\mathbf{C} \setminus \{C_i\}} P(\mathbf{c}|\mathbf{f})$. The choice of the appropriate inference to be performed given a specific loss function is further discussed in [Dembczynski *et al.*, 2012].

3 An Ensemble of Bayesian Classifiers

As with traditional classification problems, building a multilabel classifier involves a learning phase, where the training data are used to tune parameters and perform model selection, and a test phase, where the learned model (or in our case, the ensemble of models) is used to classify instances.

Our approach consists in learning an ensemble of Bayesian networks from data, and then merging these models via geometric average (logarithmic pooling) to produce classifications. Bayesian networks [Koller and Friedman, 2009] provide a compact specification for the joint probability mass function of class and feature variables. Our ensemble contains n different Bayesian networks, one for each class event C_i ($i = 1, \ldots, n$), each providing an alternative model of the joint distribution $P(\mathbf{C}, \mathbf{F})$.

3.1 Learning

The networks in the ensemble are indexed by the class events. With no lack of generality, we describe here the network associated to class C_i. First, a directed acyclic graph $\mathcal{G}_i$ over $\mathbf{C}$ such that C_i is the unique parent of all the other class events, and no more arcs are present, is considered (see Fig. 1.a). Following the Markov condition,[1] this corresponds to assume that, given C_i, the other classes are independent.

The features are assumed to be independent given the joint class $\mathbf{C}$. This extends the naive Bayes assumption to the multilabel case. At the graph level, the assumption can be accomodated by augmenting the graph $\mathcal{G}_i$, defined over $\mathbf{C}$, to a graph $\overline{\mathcal{G}}_i$ defined over the whole set of variables $(\mathbf{C}, \mathbf{F})$ as follows. The features are leaf nodes (i.e., they have no children) and their parents are, for each feature, all the class events (see Fig. 1.b).

A Bayesian network associated to $\overline{\mathcal{G}}_i$ has *maximum in-degree* (i.e., maximum number of parents) equal to the number of classes n: there are therefore conditional probability tables whose number of elements is exponential in n. This is not practical for problems with many classes. A structural learning algorithm can be therefore adopted to reduce

[1]The Markov condition for directed graphs specifies the graph semantics in Bayesian networks: any variable is conditionally independent of its non-descendant non-parents given its parents.

the maximum in-degree. In practice, to remove some class-to-feature arcs in $\overline{\mathcal{G}}_i$, we evaluate the output of the following optimization:

$$\mathcal{G}_i^* = \arg \max_{\mathcal{G}_i \subset \mathcal{G} \subset \overline{\mathcal{G}}_i} \log P(\mathcal{G}|\mathcal{D}), \tag{3}$$

where set inclusions among graphs should be intended in the arcs space, and

$$\log P(\mathcal{G}|\mathcal{D}) = \sum_{i=1}^{n} \psi_\alpha[C_i, \mathrm{Pa}(C_i)] + \sum_{j=1}^{m} \psi_\alpha[F_j, \mathrm{Pa}(F_j)], \tag{4}$$

where $\mathrm{Pa}(F_j)$ denotes the parents of F_j according to $\mathcal{G}$, and similarly $\mathrm{Pa}(C_i)$ are the parents of C_i. Moreover, ψ_α is BDEu score [Buntine, 1991] with equivalent sample size α. For instance, the score $\psi_\alpha[F_j, \mathrm{Pa}(F_j)]$ is

$$\sum_{i=1}^{|\mathrm{Pa}(F_j)|} \left[\log \frac{\Gamma(\alpha_j)}{\Gamma(\alpha_j + N_{ji})} + \sum_{k=1}^{|F_j|} \log \frac{\Gamma(\alpha_{ji} + N_{jik})}{\Gamma(\alpha_{ji})} \right], \tag{5}$$

where the second sum is over the possible states of F_j, and the first over the joint states of its parents. Moreover, N_{jik} is the number of records such that F_j is in its k-th state and its parents in their i-th configuration, while $N_{ji} = \sum_k N_{jik}$. Finally, α_{ji} is equal to α divided by the number of states of F_j and by the number of (joint) states of the parents of F_j, while $\alpha_j = \sum_i \alpha_{ji}$.

If we consider a network associated to a specific class event, the links connecting the class events are fixed; thus, the class variables have the same parents for all the graphs in the search space. This implies that the first sum in the right-hand side of (4) is constant. The optimization in (3) can be therefore achieved by considering only the features. Any subset of $\mathbf{C}$ is a candidate for the parents set of a feature, this reducing the problem to m independent local optimizations. In practice, the parents of F_j according to $\mathcal{G}^*$ are

$$\mathbf{C}_{F_j} = \arg \max_{\mathrm{Pa}(F_j) \subseteq \mathbf{C}} \psi_\alpha[F_j, \mathrm{Pa}(F_j)], \tag{6}$$

for each $j = 1, \ldots, m$. This is possible because of the bipartite separation between class events and features, while in general cases the graph maximizing all the local scores can include directed cycles. Moreover, optimization in (6) becomes more efficient by considering the pruning techniques proposed in [de Campos and Ji, 2011] (see Section 3.3).

According to Equation (6), the subgraph linking the classes does not impact on the outcome of optimization; in other words, all models of the ensemble have the same class-to-feature arcs.

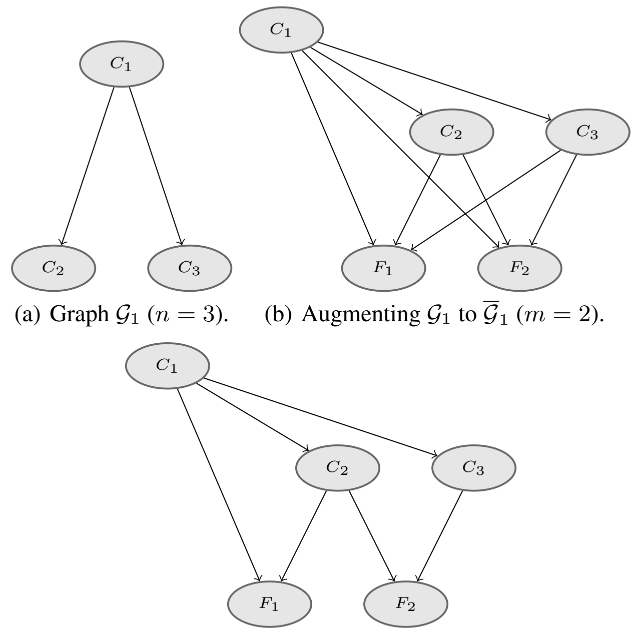
Graph $\mathcal{G}_i^*$ is the qualitative basis for the specification of the Bayesian network associated to C_i. Given the data, the network probabilities are indeed specified following a standard Bayesian learning procedure. We denote by $P_i(\mathbf{C}, \mathbf{F})$ the corresponding joint probability mass function indexed by C_i. The graph induces the following factorization:

$$P_i(\mathbf{c}, \mathbf{f}) = P(c_i) \cdot \prod_{l=1,\ldots,n, l\neq i} P(c_l|c_i) \cdot \prod_{j=1}^{m} P(f_j|\mathbf{c}_{F_j}), \tag{7}$$

with the values of $f_j, c_i, c_l, \mathbf{c}_{F_j}$ consistent with those of $\mathbf{c}, \mathbf{f}$. Note that according to the above model assumptions, the data likelihood term

$$P(\mathbf{f}|\mathbf{c}) = \prod_{j=1}^{m} P(f_j|\mathbf{c}_{F_j}) \tag{8}$$

is the same for all models in the ensemble.



(a) Graph $\mathcal{G}_1$ ($n = 3$). (b) Augmenting $\mathcal{G}_1$ to $\overline{\mathcal{G}}_1$ ($m = 2$).

(c) Removing class-to-feature arcs from $\overline{\mathcal{G}}_1$ to obtain $\mathcal{G}_1^*$.

Figure 1: The construction of the graphs of the Bayesian networks in the ensemble. The step from (b) to (c) is based on the structural learning algorithm in (6).

3.2 Classification

The models in the ensemble are combined to issue a classification according to the evaluation metric being used, in agreement with Equations (1) and (2). For global accuracy we compute the mode of the geometric average of joint distribution of each Bayesian network in the ensemble, given by

$$\hat{\mathbf{c}} = \arg \max_{\mathbf{c}} \prod_{i=1}^{n} [P_i(\mathbf{c}, \mathbf{f})]^{1/n}. \tag{9}$$

The above inference is equivalent to the computation of the MAP configuration in the Markov random field over variables $\mathbf{C}$ with potentials $\phi_{i,l}(c_i, c_l) = P_i(c_l|c_i)$, $\phi_{\mathbf{c}_{F_j}}(\mathbf{c}_{F_j}) = P(f_j|\mathbf{c}_{F_j})$ and $\phi_i(c_i) = P_i(c_i)$, $i, l = 1, \ldots, n$ ($l \neq i$) and $j = 1, \ldots, m$. Such an inference can be solved exactly by junction tree algorithms when the treewidth of the corresponding Bayesian network is sufficiently small (e.g., up to 20 in standard computers) or transformed into a mixed-integer linear program and subsequently solved by standard solvers or message-passing methods [Koller and Friedman, 2009; Sontag *et al.*, 2008]. For very large models ($n > 1000$), the problem can be solved approximately by message-passing algorithms such as TRW-S and MPLP [Kolmogorov, 2006; Globerson and Jaakkola, 2007], or yet by network flow algorithms such as extended roof duality [Rother *et al.*, 2007].

For Hamming accuracy, we instead classify an instance $\mathbf{f}$ as $\hat{\mathbf{c}} = (\hat{c}_1, \ldots, \hat{c}_n)$ such that for each $l = 1, \ldots, n$

$$\hat{c}_l = \arg\max_{c_l} \prod_{i=1}^{n} \left[P_i(c_l, \mathbf{f})\right]^{1/n}, \tag{10}$$

where

$$P_i(c_l, \mathbf{f}) = \sum_{\mathbf{C}\setminus\{C_l\}} P_i(\mathbf{c}, \mathbf{f}) \tag{11}$$

is the marginal probability of class event c_l according to model $P_i(\mathbf{c}, \mathbf{f})$.

The computation of the marginal probability distributions in (11) can also be performed by junction tree methods, when the treewidth of the corresponding Bayesian network is small, or approximated by message-passing algorithms such as belief propagation and its many variants [Koller and Friedman, 2009].

Overall we have defined two types of inference for multilabel classification, based on the models in (9) and (10), which will be called, respectively, EBN-J and EBN-M (these acronyms standing for ensemble of Bayesian networks with the joint and the marginal).

A linear, instead of log-linear, pooling could be considered to average the models. The complexity of marginal inferences is equivalent whether we use linear or log-linear pooling, and the same algorithms can be used. The theoretical complexity of MAP inference is also the same for either pooling method (their decision versions are both NP-complete, see [Park and Darwiche, 2004]), but linear pooling requires different algorithms to cope with the presence of a latent variable [Mauá and de Campos, 2012]. In our experiments, log-linear pooling produced slightly more accurate classifications and we report only these results due to the limited space.

3.3 Computational Complexity

Let us evaluate the computational complexity of the algorithms EBN-J and EBN-M presented in the previous section. We distinguish between learning and classification complexity, where the latter refers to the classification of a single instantiation of the features. Both space and time required for computations are evaluated. The orders of magnitude of these descriptors are reported as a function of the (training) dataset size k, the number of classes n, the number of features m, the average number of states for the features $f = m^{-1}\sum_{j=1}^{m} |F_j|$, and the maximum in-degree $g = \max_{j=1,\ldots,m} \|\mathbf{C}_{F_j}\|$ (where $\|\cdot\|$ returns the number elements in a joint variable). We first consider a single network in the ensemble. Regarding space, the tables $P(C_l)$, $P(C_l|C_i)$, $P(F_j|\mathbf{C}_{F_j})$ need, respectively 2, 4 and $|F_i| \cdot 2^{\|\mathbf{C}_{F_i}\|}$. We already noticed that the tables associated to the features are the same for each network. Thus, when considering the whole ensemble only the (constant) terms associated to the classes are multiplied by n and the overall space complexity is $O(4n + f2^g)$. These tables should be available during both learning and classification for both classifiers.

Regarding time, let us start from the learning. Structural learning requires $O(2^g m)$, while network quantification require to scan the dataset and learn the probabilities, i.e., for the whole ensemble, $O(nk)$. For classification, both the computation of a marginal in Bayesian network and the MAP inference in the Markov random field obtained by aggregating the models can be solved exactly by junction tree algorithms in time exponential in the network treewidth, or approximated reasonably well by message-passing algorithms in time $O(ng2^g)$. Since, as proved by [de Campos and Ji, 2011], $g = O(\log k)$, the overall complexity is polynomial (approximate inference algorithms are only used when the treewidth is too high).

3.4 Implementation

We have implemented the high-level part of the algorithm in Python. For structural learning, we modified the GOBNILP package[2] in order to consider the constraints in (3) during the search in the graphs space. The inferences necessary for classification have been performed using the junction tree and belief propagation algorithms implemented in libDAI[3], a library for inference in probabilistic graphical models.

4 Experiments

We compare our models against the binary relevance method, using naive Bayes as base classifiers (BR-NB); the ensemble of chain classifier using Naive Bayes (ECC-NB) and J48 (ECC-J48) as base classifier. ECC stands for 'ensemble of chain classifiers', where the ensemble contains a number of models which equals the number of classes. We set to 5 the number of members of the ensemble, namely the number of chains which are implemented using different random label orders. Therefore, ECC runs $5n$ base models, where n is the number of classes. We use the implementation of these methods provided by MEKA package.[4] It has not been possible comparing also with the Bayesian networks models of [Bielza *et al.*, 2011] or with the conditional random fields in [Ghamrawi and McCallum, 2005], because of the lack of public domain implementations.

Regarding the parameters of our model, we set the equivalent sample size for the structural learning to $\alpha = 5$. Moreover, we applied a bagging procedure to our ensemble, generating 5 different bootstrap replicates of the original training set. In this way, also our approach run $5n$ base models.

On each bootstrap replicate, the ensemble is learned from scratch. To combine the output of the ensemble learned on the different replicates, for EBN-J, we assign to each class the value appearing in the majority of the outputs (note that the number of bootstraps is odd). For EBN-M, instead, we take the arithmetic average of the marginals of the different bootstrapped models.

On each data set we perform feature selection using the correlation-based feature selection (CFS) [Witten *et al.*, 2011,

[2]http://www.cs.york.ac.uk/aig/sw/gobnilp

[3]http://www.libdai.org

[4]http://meka.sourceforge.net

Chap. 7.1], which is a method developed for traditional classification. We perform CFS n times, once for each different class variables. Eventually, we retain the *union* of the features selected by the different runs of CFS.

The characteristics of the data sets are in Table 1. The *density* is the average number of classes in the true state over the total number of classes. We perform 10-fold cross validation, stratifying the folds according to the most rare class. For Slashdot, given the high number instances, we validate the models by a 70-30% split.

Database	Classes	Features	Instances	Density
Emotions	6	72	593	.31
Scene	6	294	2407	.18
Yeast	14	103	2417	.30
Slashdot	22	1079	3782	.05
Genbase	27	1186	662	.04
Cal500	174	68	502	.15

Table 1: Datasets used for validation.

Since Bayesian networks need discrete features, we discretize each numerical feature into four bins. The bins are delimited by the 25-th, the 50-th and 75-th percentile of the value of the feature.

The results for the different data sets are provided in Tables 2–7. We report both global accuracy (acc) and Hamming accuracy (H_{acc}). Regarding our ensembles, we evaluate EBN-J in terms of global accuracy and EBN-M in terms of Hamming accuracy. Thus, we perform a different inference depending on the accuracy function which has to be maximized. Empirical tests confirmed that, as expected (see the discussion in Section 3.2), EBN-J has higher global accuracy and lower Hamming accuracy than EBN-M. In the special case of the Cal500 data set, the very high number of classes makes unnecessary the evaluation of the global accuracy which is almost zero for all the algorithms. Only Hamming accuracies are therefore reported.

Regarding computation times, EBN-J and EBN-M are one or two orders of magnitude slower than the MEKA implementations of ECC-J48/NB (which, in turn, are slower than BR-NB). This gap is partially due to our implementation, which at this stage is only prototypal.

We summarize the results in Table 8, which provides the mean rank of the various classifiers across the various data sets. For each indicator of performance, the best-performing approach is boldfaced. Although we cannot claim statistical significance of the difference among ranks (more data sets need to be analyzed to this purpose), the proposed ensemble achieves the best rank on both metrics; it is clearly competitive with state-of-the art approaches.

5 Conclusions

A novel approach to multilabel classification with Bayesian networks is proposed, based on an ensemble of models averaged by logarithmic opinion pool. Two different inference algorithm based respectively on the MAP solution on the joint model (EBN-J) and on the average of the marginals (EBN-M)

Algorithm	acc	H_{acc}
BR-NB	$.261 \pm .049$	$.776 \pm .023$
ECC-NB	$\mathbf{.295 \pm .060}$	$\mathbf{.781 \pm .026}$
ECC-J48	$.260 \pm .038$	$.780 \pm .027$
EBN-J	$.263 \pm .062$	–
EBN-M	–	$.780 \pm .022$

Table 2: Results for the Emotions dataset.

Algorithm	acc	H_{acc}
BR-NB	$.276 \pm .017$	$.826 \pm .008$
ECC-NB	$.294 \pm .022$	$.835 \pm .007$
ECC-J48	$.531 \pm .038$	$\mathbf{.883 \pm .008}$
EBN-J	$\mathbf{.575 \pm .030}$	–
EBN-M	–	$.880 \pm .010$

Table 3: Results for the Scene dataset.

Algorithm	acc	H_{acc}
BR-NB	$.091 \pm .020$	$0.703 \pm .011$
ECC-NB	$.102 \pm .023$	$.703 \pm .009$
ECC-J48	$\mathbf{.132 \pm .023}$	$.771 \pm .007$
EBN-J	$.127 \pm .018$	–
EBN-M	–	$\mathbf{.773 \pm .008}$

Table 4: Results for the Yeast dataset.

Algorithm	acc	H_{acc}
BR-NB	.361	.959
ECC-NB	.368	.959
ECC-J48	.370	.959
EBN-J	**.385**	–
EBN-M	–	.959

Table 5: Results for the Slashdot dataset.

Algorithm	acc	H_{acc}
BR-NB	$.897 \pm .031$	$.996 \pm .001$
ECC-NB	$.897 \pm .031$	$.996 \pm .001$
ECC-J48	$.934 \pm .015$	$.998 \pm .001$
EBN-J	$\mathbf{.965 \pm .015}$	–
EBN-M	–	$.998 \pm .001$

Table 6: Results for the Genbase dataset.

Algorithm	H_{acc}
BR-NB	$.615 \pm .011$
ECC-NB	$.570 \pm .012$
ECC-J48	$.614 \pm .008$
EBN-M	$\mathbf{.859 \pm .003}$

Table 7: Results for the Cal500 dataset.

Algorithm	acc	H_{acc}
BR-NB	3.2	3.0
ECC-NB	2.70	2.7
ECC-J48	2.4	2.2
EBN-J	**1.7**	–
EBN-M	–	**2.1**

Table 8: Mean rank of the models over the various data sets.

are proposed, respectively to maximize the exact match and the Hamming accuracy. Empirical validation shows competing results with the state of the art. As future work we intend to support missing data both in the learning and in the classification step. Also more sophisticated mixtures could be considered, for instance by learning the maximum a posteriori weights for the mixture of the models.

Acknowledgments

We thank Cassio P. de Campos and David Huber for valuable discussions.

References

[Bielza *et al.*, 2011] C. Bielza, G. Li, and P. Larranaga. Multi-dimensional classification with Bayesian networks. *International Journal of Approximate Reasoning*, 52(6):705–727, 2011.

[Buntine, 1991] W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the 8th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 52–60, 1991.

[de Campos and Ji, 2011] C.P. de Campos and Q. Ji. Efficient structure learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689, 2011.

[Dembczynski *et al.*, 2010] K. Dembczynski, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 279–286, Haifa, Israel, 2010.

[Dembczynski *et al.*, 2012] K. Dembczynski, W. Waegeman, and E. Hüllermeier. An analysis of chaining in multilabel classification. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI)*, pages 294–299, 2012.

[Ghamrawi and McCallum, 2005] N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 195–200, New York, NY, USA, 2005.

[Globerson and Jaakkola, 2007] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Advances in Neural Information Processing Systems 20 (NIPS)*, 2007.

[Heskes, 1997] T. Heskes. Selecting weighting factors in logarithmic opinion pools. In *Advances in Neural Information Processing Systems 10 (NIPS)*, 1997.

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

[Kolmogorov, 2006] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1568–1583, 2006.

[Mauá and de Campos, 2012] D. D. Mauá and C. P. de Campos. Anytime marginal map inference. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2012)*, 2012.

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

[Read *et al.*, 2011] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333–359, 2011.

[Rother *et al.*, 2007] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, june 2007.

[Sontag *et al.*, 2008] D. Sontag, T. Meltzer, A. Globerson, Y. Weiss, and T. Jaakkola. Tightening LP relaxations for MAP using message-passing. In *24th Conference in Uncertainty in Artificial Intelligence*, pages 503–510. AUAI Press, 2008.

[Timmermann, 2006] A. Timmermann. Forecast combinations. *Handbook of economic forecasting*, 1:135–196, 2006.

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

[Witten *et al.*, 2011] I.H. Witten, E. Frank, and M.A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2011.

[Yang *et al.*, 2007] Y. Yang, G.I. Webb, J. Cerquides, K.B. Korb, J. Boughton, and K.M. Ting. To select or to weigh: A comparative study of linear combination schemes for superparent-one-dependence estimators. *IEEE Transactions on Knowledge and Data Engineering*, 19(12):1652–1665, 2007.