A hybrid algorithm for Bayesian network structure learning with application to multi-label learning
Maxime Gasse, Alex Aussem, Haytham Elghazel

To cite this version:

HAL Id: hal-01153255
https://hal.archives-ouvertes.fr/hal-01153255
Submitted on 19 May 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives| 4.0 International License
A hybrid algorithm for Bayesian network structure learning with application to multi-label learning

Maxime Gasse, Alex Aussem*, Haytham Elghazel

Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205, F-69622, France

Abstract

We present a novel hybrid algorithm for Bayesian network structure learning, called H2PC. It first reconstructs the skeleton of a Bayesian network and then performs a Bayesian-scoring greedy hill-climbing search to orient the edges. The algorithm is based on divide-and-conquer constraint-based subroutines to learn the local structure around a target variable. We conduct two series of experimental comparisons of H2PC against Max-Min Hill-Climbing (MMHC), which is currently the most powerful state-of-the-art algorithm for Bayesian network structure learning. First, we use eight well-known Bayesian network benchmarks with various data sizes to assess the quality of the learned structure returned by the algorithms. Our extensive experiments show that H2PC outperforms MMHC in terms of goodness of fit to new data and quality of the network structure with respect to the true dependence structure of the data. Second, we investigate H2PC’s ability to solve the multi-label learning problem. We provide theoretical results to characterize and identify graphically the so-called minimal label powersets that appear as irreducible factors in the joint distribution under the faithfulness condition. The multi-label learning problem is then decomposed into a series of multi-class classification problems, where each multi-class variable encodes a label powerset. H2PC is shown to compare favorably to MMHC in terms of global classification accuracy over ten multi-label data sets covering different application domains. Overall, our experiments support the conclusions that local structural learning with H2PC in the form of local neighborhood induction is a theoretically well-motivated and empirically effective learning framework that is well suited to multi-label learning. The source code (in R) of H2PC as well as all data sets used for the empirical tests are publicly available.

Keywords: Bayesian networks, Multi-label learning, Markov boundary, Feature subset selection.

1. Introduction

A Bayesian network (BN) is a probabilistic model formed by a structure and parameters. The structure of a BN is a directed acyclic graph (DAG), whilst its parameters are conditional probability distributions associated with the variables in the model. The problem of finding the DAG that encodes the conditional independencies present in the data attracted a great deal of interest over the last years (Rodrigues de Morais & Aussem, 2010a; Scutari, 2010; Scutari & Brogini, 2012; Kojima et al., 2010; Perrier et al., 2008; Villanueva & Maciel, 2012; Peña, 2010; Gasse et al., 2012). The inferred DAG is very useful for many applications, including feature selection (Aliferis et al., 2010; Peña et al., 2007; Rodrigues de Morais & Aussem, 2010b), causal relationships inference from observational data (Ellis & Wong, 2008; Aliferis et al., 2010; Aussem et al., 2012, 2010; Prestat et al., 2013; Cawley, 2008; Brown & Tsamardinos, 2008) and more recently multi-label learning (Dembczyński et al., 2012; Zhang & Zhang, 2010; Guo & Gu, 2011).

Ideally the DAG should coincide with the dependence structure of the global distribution, or it should at least identify a distribution as close as possible to the correct one in the probability space. This step, called structure learning, is similar in approaches and terminology to model selection procedures for classical statistical models. Basically, constraint-based (CB) learning methods systematically check the data for conditional independence relationships and use them as constraints to construct a partially oriented graph representative of a BN.
equivalence class, whilst search-and-score (SS) methods make use of a goodness-of-fit score function for evaluating graphical structures with regard to the data set. Hybrid methods attempt to get the best of both worlds: they learn a skeleton with a CB approach and constrain on the DAGs considered during the SS phase.

In this study, we present a novel hybrid algorithm for Bayesian network structure learning, called H2PC. It first reconstructs the skeleton of a Bayesian network and then performs a Bayesian-scoring greedy hill-climbing search to orient the edges. The algorithm is based on divide-and-conquer constraint-based subroutines to learn the local structure around a target variable. HPC may be thought of as a way to compensate for the large number of false negatives at the output of the weak PC learner, by performing extra computations. As this may arise at the expense of the number of false positives, we control the expected proportion of false discoveries (i.e. false positive nodes) among all the discoveries made in PC. We use a modification of the Incremental association Markov boundary algorithm (IAMB), initially developed by Tsamardinos et al. in (Tsamardinos et al., 2003) and later modified by Jose Peña in (Peña, 2008) to control the FDR of edges when learning Bayesian network models. HPC scales to thousands of variables and can deal with many fewer samples \((n < q)\). To illustrate its performance by means of empirical evidence, we conduct two series of experimental comparisons of H2PC against Max-Min Hill-Climbing (MMHC), which is currently the most powerful state-of-the-art algorithm for BN structure learning (Tsamardinos et al., 2006), using well-known BN benchmarks with various data sizes, to assess the goodness of fit to new data as well as the quality of the network structure with respect to the true dependence structure of the data.

We then address a real application of H2PC where the true dependence structure is unknown. More specifically, we investigate H2PC’s ability to encode the joint distribution of the label set conditioned on the input features in the multi-label classification (MLC) problem. Many challenging applications, such as photo and video annotation and web page categorization, can benefit from being formulated as MLC tasks with large number of categories (Dembczynski et al., 2012; Read et al., 2009; Madjarov et al., 2012; Kocev et al., 2007; Tsoumakas et al., 2010b). Recent research in MLC focuses on the exploitation of the label conditional dependency in order to better predict the label combination for each example. We show that local BN structure discovery methods offer an elegant and powerful approach to solve this problem. We establish two theorems (Theorem 6 and 7) linking the concepts of marginal Markov boundaries, joint Markov boundaries and so-called label powersets under the faithfulness assumption. These Theorems offer a simple guideline to characterize graphically: i) the minimal label powerset decomposition, (i.e. into minimal subsets \(Y_{LP} \subseteq Y\) such that \(Y_{LP} \perp \ Y \setminus Y_{LP} \mid X\)), and ii) the minimal subset of features, w.r.t an Information Theory criterion, needed to predict each label powerset, thereby reducing the input space and the computational burden of the multi-label classification. To solve the MLC problem with BNs, the DAG obtained from the data plays a pivotal role. So in this second series of experiments, we assess the comparative ability of H2PC and MMHC to encode the label dependency structure by means of an indirect goodness of fit indicator, namely the 0/1 loss function, which makes sense in the MLC context.

The rest of the paper is organized as follows: In the Section 2, we review the theory of BN and discuss the main BN structure learning strategies. We then present the H2PC algorithm in details in Section 3. Section 4 evaluates our proposed method and shows results for several tasks involving artificial data sampled from known BNs. Then we report, in Section 5, on our experiments on real-world data sets in a multi-label learning context so as to provide empirical support for the proposed methodology. The main theoretical results appear formally as two theorems (Theorem 8 and 9) in Section 5. Their proofs are established in the Appendix. Finally, Section 6 raises several issues for future work and we conclude in Section 7 with a summary of our contribution.

2. Preliminaries

We define next some key concepts used along the paper and state some results that will support our analysis. In this paper, upper-case letters in italics denote random variables (e.g., \(X, Y\)) and lower-case letters in italics denote their values (e.g., \(x, y\)). Upper-case bold letters denote random variable sets (e.g., \(X, Y, Z\)) and lower-case bold letters denote their values (e.g., \(x, y\)). We denote by \(X \perp \ Y \mid Z\) the conditional independence between \(X\) and \(Y\) given the set of variables \(Z\). To keep the notation uncluttered, we use \(p(y \mid x)\) to denote \(p(Y = y \mid X = x)\).

---

1 A first version of HP2C without FDR control has been discussed in a paper that appeared in the Proceedings of ECML-PKDD, pages 58-73, 2012.
2.1. Bayesian networks

Formally, a BN is a tuple \(< \mathcal{G}, P >\), where \( \mathcal{G} = (\mathbf{U}, \mathbf{E}) \) is a directed acyclic graph (DAG) with nodes representing the random variables \( \mathbf{U} \) and \( P \) a joint probability distribution in \( \mathcal{U} \). In addition, \( \mathcal{G} \) and \( P \) must satisfy the Markov condition: every variable, \( X \in \mathbf{U} \), is independent of any subset of its non-descendant variables conditioned on the set of its parents, denoted by \( \text{Pa}^G \). From the Markov condition, it is easy to prove (Neapolitan, 2004) that the joint probability distribution \( P \) on the variables in \( \mathbf{U} \) can be factored as follows:

\[
P(\mathbf{V}) = P(\mathbf{X}_1, \ldots, \mathbf{X}_n) = \prod_{i=1}^n P(X_i | \text{Pa}^G_i)
\]

Equation 1 allows a parsimonious decomposition of the joint distribution \( P \). It enables us to reduce the problem of determining a huge number of probability values to that of determining relatively few.

A BN structure \( \mathcal{G} \) entails a set of conditional independence assumptions. They can all be identified by the \( d \)-separation criterion (Pearl, 1988). We use \( X \perp \!
\perp Y | Z \) to denote the assertion that \( X \) is \( d \)-separated from \( Y \) given \( Z \) in \( \mathcal{G} \). Formally, \( X \perp \!
\perp Y | Z \) is true when for every undirected path in \( \mathcal{G} \) between \( X \) and \( Y \), there exists a node \( W \) in the path such that either (1) \( W \) does not have two parents in the path and \( W \in Z \), or (2) \( W \) has two parents in the path and neither \( W \) nor its descendants is in \( Z \). If \( < \mathcal{G}, P > \) is a BN, \( X \perp \!
\perp Y | Z \) if \( X \perp \!
\perp Y | Z \). The converse does not necessarily hold. We say that \( < \mathcal{G}, P > \) satisfies the faithfulness condition if the \( d \)-separations in \( \mathcal{G} \) identify all and only the conditional independencies in \( P \), i.e., \( X \perp \!
\perp Y | Z \) iff \( X \perp \!
\perp Y | Z \).

A Markov blanket \( \mathbf{M}_T \) of \( T \) is any set of variables such that \( T \) is conditionally independent of all the remaining variables given \( \mathbf{M}_T \). By extension, a Markov blanket of \( T \) in \( \mathbf{V} \) guarantees that \( \mathbf{M}_T \subseteq \mathbf{V} \), and that \( T \) is conditionally independent of the remaining variables in \( \mathbf{V} \). A Markov boundary, \( \mathbf{MB}_T \), of \( T \) is any Markov blanket such that none of its proper subsets is a Markov blanket of \( T \).

We denote by \( \text{PC}^G \), the set of parents and children of \( T \) in \( \mathcal{G} \), and by \( \text{SP}^G \), the set of spouses of \( T \) in \( \mathcal{G} \). The spouses of \( T \) are the variables that have common children with \( T \). These sets are unique for all \( \mathcal{G} \), such that \( < \mathcal{G}, P > \) satisfies the faithfulness condition and so we will drop the superscript \( G \). We denote by \( \text{dSep}(X) \), the set that \( d \)-separates \( X \) from (the implicit) target \( T \).

**Theorem 1.** Suppose \( < \mathcal{G}, P > \) satisfies the faithfulness condition. Then \( X \) and \( Y \) are not adjacent in \( \mathcal{G} \) iff \( \exists Z \in \mathbf{U} \setminus \{X,Y\} \) such that \( X \perp \!
\perp Y | Z \). Moreover, \( \mathbf{MB}_X = \mathbf{PC}_X \cup \mathbf{SP}_X \).

A proof can be found for instance in (Neapolitan, 2004).

Two graphs are said equivalent iff they encode the same set of conditional independencies via the \( d \)-separation criterion. The equivalence class of a DAG \( \mathcal{G} \) is a set of DAGs that are equivalent to \( \mathcal{G} \). The next result showed by (Pearl, 1988), establishes that equivalent graphs have the same undirected graph but might disagree on the direction of some of the arcs.

**Theorem 2.** Two DAGs are equivalent iff they have the same underlying undirected graph and the same set of v-structures (i.e. converging edges into the same node, such as \( X \rightarrow Y \leftarrow Z \)).

Moreover, an equivalence class of network structures can be uniquely represented by a partially directed DAG (PDAG), also called a DAG pattern. The DAG pattern is defined as the graph that has the same links as the DAGs in the equivalence class and has oriented all and only the edges common to the DAGs in the equivalence class.

A structure learning algorithm from data is said to be correct (or sound) if it returns the correct DAG pattern (or a DAG in the correct equivalence class) under the assumptions that the independence tests are reliable and that the learning database is a sample from a distribution \( P \) faithful to a DAG \( \mathcal{G} \). The (ideal) assumption that the independence tests are reliable means that they decide (in)dependence iff the (in)dependence holds in \( P \).

2.2. Conditional independence properties

The following three theorems, borrowed from Peña et al. (2007), are proven in Pearl (1988):

**Theorem 3.** Let \( \mathbf{X}, \mathbf{Y}, \mathbf{Z} \) and \( \mathbf{W} \) denote four mutually disjoint subsets of \( \mathbf{U} \). Any probability distribution \( p \) satisfies the following four properties: Symmetry \( X \perp \!
\perp Y | Z \Rightarrow Y \perp \!
\perp X | Z \), Decomposition, \( X \perp \!
\perp (Y \cup W) | Z \Rightarrow X \perp \!
\perp Y | Z \), Weak Unchain \( X \perp \!
\perp (Y \cup W) | Z \Rightarrow X \perp \!
\perp Y | (Z \cup W) \), and Contraction, \( X \perp \!
\perp Y | (Z \cup W) \Rightarrow X \perp \!
\perp W | Z \Rightarrow X \perp \!
\perp (Y \cup W) | Z \).

**Theorem 4.** If \( p \) is strictly positive, then \( p \) satisfies the previous four properties plus the Intersection property \( X \perp \!
\perp Y | (Z \cup W) \Rightarrow X \perp \!
\perp W | Z \Rightarrow X \perp \!
\perp (Y \cup W) | Z \). Additionally, each \( X \in \mathbf{U} \) has a unique Markov boundary, \( \mathbf{MB}_X \).

**Theorem 5.** If \( p \) is faithful to a DAG \( \mathcal{G} \), then \( p \) satisfies the previous five properties plus the Composition property \( X \perp \!
\perp Y | Z \Rightarrow X \perp \!
\perp (Y \cup W) | Z \) and the local Markov property \( X \perp \!
\perp (\text{ND}_X \cup \text{Pa}_X) | \text{Pa}_X \) for each \( X \in \mathbf{U} \), where \( \text{ND}_X \) denotes the non-descendants of \( X \) in \( \mathcal{G} \).
2.3. Structure learning strategies

The number of DAGs, \( \mathcal{G} \), is super-exponential in the number of random variables in the domain and the problem of learning the most probable a posteriori BN from data is worst-case NP-hard (Chickering et al., 2004). One needs to resort to heuristical methods in order to be able to solve very large problems effectively.

Both CB and SS heuristic approaches have advantages and disadvantages. CB approaches are relatively quick, deterministic, and have a well defined stopping criterion; however, they rely on an arbitrary significance level to test for independence, and they can be unstable in the sense that an error early on in the search can have a cascading effect that causes many errors to be present in the final graph. SS approaches have the advantage of being able to flexibly incorporate users’ background knowledge in the form of prior probabilities over the structures and are also capable of dealing with incomplete records in the database (e.g. EM technique). Although SS methods are favored in practice when dealing with small dimensional data sets, they are slow to converge and the computational complexity often prevents us from finding optimal BN structures (Perrier et al., 2008; Kojima et al., 2010). With currently available exact algorithms (Koivisto & Sood, 2004; Slander & Myllymäki, 2006; Cussens & Bartlett, 2013; Studený & Haws, 2014) and a decomposable score like BDeu, the computational complexity remains exponential, and therefore, such algorithms are intractable for BNs with more than around 30 vertices on current workstations. For larger sets of variables the computational burden becomes prohibitive and restrictions about the structure have to be imposed, such as a limit on the size of the parent sets. With this in mind, the ability to restrict the search locally around the target variable is a key advantage of CB methods over SS methods. They are able to construct a local graph around the target node without having to construct the whole BN first, hence their scalability (Peña et al., 2007; Rodrigues de Morais & Aussem, 2010b,a; Tsamardinos et al., 2006; Peña, 2008).

With a view to balancing the computation cost with the desired accuracy of the estimates, several hybrid methods have been proposed recently. Tsamardinos et al. proposed in (Tsamardinos et al., 2006) the Min-Max Hill Climbing (MMHC) algorithm and conducted one of the most extensive empirical comparison performed in recent years showing that MMHC was the fastest and the most accurate method in terms of structural error based on the structural hamming distance. More specifically, MMHC outperformed both in terms of time efficiency and quality of reconstruction the PC (Spirtes et al., 2000), the Sparse Candidate (Friedman et al., 1999b), the Three Phase Dependency Analysis (Cheng et al., 2002), the Optimal Reinsertion (Moore & Wong, 2003), the Greedy Equivalence Search (Chickering, 2002), and the Greedy Hill-Climbing Search on a variety of networks, sample sizes, and parameter values. Although MMHC is rather heuristic by nature (it returns a local optimum of the score function), it is currently considered as the most powerful state-of-the-art algorithm for BN structure learning capable of dealing with thousands of nodes in reasonable time.

In order to enhance its performance on small dimensional data sets, Perrier et al. proposed in (Perrier et al., 2008) a hybrid algorithm that can learn an optimal BN (i.e., it converges to the true model in the sample limit) when an undirected graph is given as a structural constraint. They defined this undirected graph as a super-structure (i.e., every DAG considered in the SS phase is compelled to be a subgraph of the super-structure). This algorithm can learn optimal BNs containing up to 50 vertices when the average degree of the super-structure is around two, that is, a sparse structural constraint is assumed. To extend its feasibility to BN with a few hundred of vertices and an average degree up to four, Kojima et al. proposed in (Kojima et al., 2010) to divide the super-structure into several clusters and perform an optimal search on each of them in order to scale up to larger networks. Despite interesting improvements in terms of score and structural hamming distance on several benchmark BNs, they report running times about 10^4 times longer than MMHC on average, which is still prohibitive.

Therefore, there is great deal of interest in hybrid methods capable of improving the structural accuracy of both CB and SS methods on graphs containing up to thousands of vertices. However, they make the strong assumption that the skeleton (also called super-structure) contains at least the edges in the true network and as small as possible extra edges. While controlling the false discovery rate (i.e., false extra edges) in BN learning has attracted some attention recently (Armen & Tsamardinos, 2011; Peña, 2008; Tsamardinos & Brown, 2008), to our knowledge, there is no work on controlling actively the rate of false-negative errors (i.e., false missing edges).

2.4. Constraint-based structure learning

Before introducing the H2PC algorithm, we discuss the general idea behind CB methods. The induction of local or global BN structures is handled by CB methods through the identification of local neighborhoods (i.e., \( \mathcal{PC}_X \)), hence their scalability to very high dimensional
data sets. CB methods systematically check the data for conditional independence relationships in order to infer a target’s neighborhood. Typically, the algorithms run either a $G^2$ or a $\chi^2$ independence test when the data set is discrete and a Fisher’s $Z$ test when it is continuous in order to decide on dependence or independence, that is, upon the rejection or acceptance of the null hypothesis of conditional independence. Since we are limiting ourselves to discrete data, both the global and the local distributions are assumed to be multinomial, and the latter are represented as conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables. Conditional independence tests and network scores for discrete data are functions of these conditional probability tables.

### 3. The H2PC algorithm

In this section, we present our hybrid algorithm for Bayesian network structure learning, called Hybrid HPC (H2PC). It first reconstructs the skeleton of a Bayesian network and then performs a Bayesian-scoring greedy hill-climbing search to filter and orient the edges. It is based on a CB subroutine called HPC to learn the parents and children of a single variable. So, we shall discuss HPC first and then move to H2PC.

#### 3.1. Parents and Children Discovery

HPC (Algorithm 1) can be viewed as an ensemble method for combining many weak PC learners in an attempt to produce a stronger PC learner. HPC is based on three subroutines: Data-Efficient Parents and Children Superset (DE-PCS), Data-Efficient Spouses Superset (DE-SPS), and Incremental Association Parents and Children with false discovery rate control (FDR-IAPC), a weak PC learner based on FDR-IAMB (Peña, 2008) that requires little computation. HPC receives a target node $T$, a data set $D$ and a set of variables $U$ as input and returns an estimation of $PC_T$. It is hybrid in that it combines the benefits of incremental and divide-and-conquer methods. The procedure starts by extracting a superset $PCS_T$ of $PC_T$ (line 1) and a superset $SPS_T$ of $SP_T$ (line 2) with a severe restriction on the maximum conditioning size ($Z = z$) in order to significantly increase the reliability of the tests. A candidate PC set is then obtained by running the weak PC learner called FDR-IAPC on $PCS_T \cup SPS_T$ (line 3). The key idea is the decentralized search at lines 4-8 that includes, in the candidate PC set, all variables in the superset $PCS_T \setminus PC_T$ that have $T$ in their vicinity. So, HPC may be thought of as a way to compensate for the large number of false negatives at the output of the weak PC learner, by performing extra computations. Note that, in theory, $X$ is in the output of FDR-IAPC($Y$) if and only if $Y$ is in the output of FDR-IAPC($X$). However, in practice, this may not always be true, particularly when working in high-dimensional domains (Peña, 2008). By loosening the criteria by which two nodes are said adjacent, the effective restrictions on the size of the neighborhood are now far less severe. The decentralized search has a significant impact on the accuracy of HPC as we shall see in in the experiments. We proved in
(Rodrigues de Morais & Aussem, 2010a) that the original HPC(T) is consistent, i.e. its output converges in probability to PC_T, if the hypothesis tests are consistent. The proof also applies to the modified version presented here.

We now discuss the subroutines in more detail. FDR-IAPC (Algorithm 2) is a fast incremental method that receives a data set D and a target node T as its input and promptly returns a rough estimation of PC_T, hence the term “weak” PC learner. In this study, we use FDR-IAPC because it aims at controlling the expected proportion of false discoveries (i.e., false positive nodes in PC_T) among all the discoveries made. FDR-IAPC is a straightforward extension of the algorithm IAMBFDR developed by Jose Peña in (Peña, 2008), which is itself a modification of the incremental association Markov boundary algorithm (IAMB) (Tsamardinos et al., 2003), to control the expected proportion of false discoveries (i.e., false positive nodes) in the estimated Markov boundary. FDR-IAPC simply removes, at lines 3-6, the spouses SP_T from the estimated Markov boundary MB_T output by IAMBFDR at line 1, and returns PC_T assuming the faithfulness condition.

The subroutines DE-PCS (Algorithm 3) and DE-SPS (Algorithm 4) search a superset of PC_T and SP_T respectively with a severe restriction on the maximum conditioning size (|Z| <= 1 in DE-PCS and |Z| <= 2 in DE-SPS) in order to significantly increase the reliability of the tests. The variable filtering has two advantages: i) it allows HPC to scale to hundreds of thousands of variables by restricting the search to a subset of relevant variables, and ii) it eliminates many true or approximate deterministic relationships that produce many false negative errors in the output of the algorithm, as explained in (Rodrigues de Morais & Aussem, 2010b,a). DE-SPS works in two steps. First, a growing phase (lines 4-8) adds the variables that are d-separated from the target but still remain associated with the target when conditioned on another variable from PCS_T. The shrinking phase (lines 9-16) discards irrelevant variables that are ancestors or descendants of a target’s spouse. Pruning such irrelevant variables speeds up HPC.

3.2. Hybrid HPC (H2PC)

In this section, we discuss the SS phase. The following discussion draws strongly on (Tsamardinos et al., 2006) as the SS phase in Hybrid HPC and MMHC are exactly the same. The idea of constraining the search to improve time-efficiency first appeared in the Sparse Candidate algorithm (Friedman et al., 1999b). It results in efficiency improvements over the (unconstrained) greedy search. All recent hybrid algorithms

---

**Algorithm 1 HPC**

Require: T: target; D: data set; U: set of variables

Ensure: PC_T: Parents and Children of T

1: [PCS_T, dSep] ← DE-PCS(T, D, U)  
2: SPS_T ← DE-SPS(T, D, U, PCS_T, dSep)  
3: PC_T ← FDR-IAPC(T, D, (T ∪ PCS_T ∪ SPS_T))  
4: for all X ∈ PCS_T \ PC_T do  
5: if ∃ T ∈ FDR-IAPC(X, D, (T ∪ PCS_T ∪ SPS_T)) then  
6: PC_T ← PC_T \ X  
7: end if  
8: end for

---

**Algorithm 2 FDR-IAPC**

Require: T: target; D: data set; U: set of variables

Ensure: PC_T: Parents and children of T;

* Learn the Markov boundary of T

1: MB_T ← IAMBFDR(X, D, U)  
2: PC_T ← MB_T  
3: for all X ∈ MB_T do  
4: if ∃ Z ⊆ (MB_T \ X) such that T ⊥ X | Z then  
5: PC_T ← PC_T \ X  
6: end if  
7: end for

---

**Algorithm 3 DE-PCS**

Require: T: target; D: data set; U: set of variables;

Ensure: PCS_T: parents and children superset of T; dSep: d-separating sets;

Phase I: Remove X if T ⊥ X

1: PCS_T ← U \ T  
2: for all X ∈ PCS_T do  
3: if (T ⊥ X) then  
4: PCS_T ← PCS_T \ X  
5: dSep(X) ← ∅  
6: end if  
7: end for

Phase II: Remove X if T ⊥ X|Y

8: for all X ∈ PCS_T do  
9: for all Y ∈ PCS_T \ X do  
10: if (T ⊥ X | Y) then  
11: PCS_T ← PCS_T \ X  
12: dSep(X) ← Y  
13: break loop FOR  
14: end if  
15: end for  
16: end for
Algorithm 4 \textit{DE-SPS}

\textbf{Require:} \(T\): target; \(D\): data set; \(U\): the set of variables; 
\(PCS_T\): parents and children superset of \(T\); \(dSep\): \(d\)-separating sets; 

\textbf{Ensure:} \(SPS_T\): Superset of the spouses of \(T\):

1: \(SPS_T \leftarrow \emptyset\)
2: \(\text{for all } X \in PCS_T \text{ do} \)
3: \(SPS_X^T \leftarrow \emptyset\)
4: \(\text{for all } Y \in U \setminus \{T \cup PCS_T\} \text{ do} \)
5: \(\text{if } (T \perp dSep(Y) \cup X) \text{ then} \)
6: \(SPS_X^T \leftarrow SPS_X^T \cup Y\)
7: \(\text{end if} \)
8: \(\text{end for} \)
9: \(\text{for all } Y \in SPS_X^T \text{ do} \)
10: \(\text{for all } Z \in SPS_X^T \setminus Y \text{ do} \)
11: \(\text{if } (T \perp Y|X \cup Z) \text{ then} \)
12: \(SPS_X^T \leftarrow SPS_X^T \setminus Y\)
13: \(\text{break loop FOR} \)
14: \(\text{end if} \)
15: \(\text{end for} \)
16: \(\text{end for} \)
17: \(SPS_T \leftarrow SPS_T \cup SPS_X^T\)
18: \(\text{end for} \)

Algorithm 5 \textit{Hybrid HPC}

\textbf{Require:} \(D\): data set; \(U\): set of variables  

\textbf{Ensure:} A DAG \(\mathcal{G}\) on the variables \(U\):

1: \(\text{for all pair of nodes } X, Y \in U \text{ do} \)
2: \(\text{Add } X \text{ in } PC_Y \text{ and Add } Y \text{ in } PC_X \text{ if } X \in HPC(Y) \text{ and } Y \in HPC(X) \)
3: \(\text{end for} \)
4: \(\text{Starting from an empty graph, perform greedy hill-climbing with operators add-edge, delete-edge, reverse-edge. Only try operator add-edge } X \rightarrow Y \text{ if } Y \in PC_X \)

build on this idea, but employ a sound algorithm for identifying the candidate parent sets. The Hybrid HPC first identifies the parents and children set of each variable, then performs a greedy hill-climbing search in the space of BN. The search begins with an empty graph. The edge addition, deletion, or direction reversal that leads to the largest increase in score (the BDeu score with uniform prior was used) is taken and the search continues in a similar fashion recursively. The important difference from standard greedy search is that the search is constrained to only consider adding an edge if it was discovered by HPC in the first phase. We extend the greedy search with a TABU list (Friedman et al., 1999b). The list keeps the last 100 structures explored. Instead of applying the best local change, the best local change that results in a structure not on the list is performed in an attempt to escape local maxima. When 15 changes occur without an increase in the maximum score ever encountered during search, the algorithm terminates. The overall best scoring structure is then returned. Clearly, the more false positives the heuristic allows to enter candidate PC set, the more computational burden is imposed in the SS phase.

4. Experimental validation on synthetic data

Before we proceed to the experiments on real-world multi-label data with H2PC, we first conduct an experimental comparison of H2PC against MMHC on synthetic data sets sampled from eight well-known benchmarks with various data sizes in order to gauge the practical relevance of the H2PC. These BNs that have been previously used as benchmarks for BN learning algorithms (see Table 1 for details). Results are reported in terms of various performance indicators to investigate how well the network generalizes to new data and how well the learned dependence structure matches the true structure of the benchmark network. We implemented H2PC in \(R\) (R Core Team, 2013) and integrated the code into the \textit{bnlearn} package from (Scutari, 2010). MMHC was implemented by Marco Scutari in \textit{bnlearn}. The source code of H2PC as well as all data sets used for the empirical tests are publicly available\(^2\). The threshold considered for the type I error of the test is 0.05. Our experiments were carried out on PC with Intel(R) Core(TM) i5-3470M CPU @3.20 GHz 4Go RAM running under Linux 64 bits.

We do not claim that those data sets resemble real-world problems, however, they make it possible to com-

\(^2\)https://github.com/madbix/bnlearn-clone-3.4
describe below: the SS phase, we report the five performance indicators
that assess the quality of the final DAG output at the end of
call, as proposed in (Peña et al., 2007). Second, to as-
semble the outputs of the algorithms with the known struc-
ture. All BN benchmarks (structure and probability ta-
bles) were downloaded from the
http://www.bnlearn.com/bnrepository
3

Table 1: Description of the BN benchmarks used in the experiments.

<table>
<thead>
<tr>
<th>network</th>
<th># var.</th>
<th># edges</th>
<th>max. degree</th>
<th>domain range</th>
<th>min/med/max PC set size</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>20</td>
<td>25</td>
<td>2/7</td>
<td>2-6</td>
<td>1/2/8</td>
</tr>
<tr>
<td>insurance</td>
<td>27</td>
<td>52</td>
<td>3/7</td>
<td>2-5</td>
<td>1/3/9</td>
</tr>
<tr>
<td>mildew</td>
<td>35</td>
<td>46</td>
<td>3/3</td>
<td>3-100</td>
<td>1/2/5</td>
</tr>
<tr>
<td>alarm</td>
<td>37</td>
<td>46</td>
<td>4/5</td>
<td>2-4</td>
<td>1/2/6</td>
</tr>
<tr>
<td>halffinder</td>
<td>56</td>
<td>66</td>
<td>4/16</td>
<td>2-11</td>
<td>1/1.5/17</td>
</tr>
<tr>
<td>minni1</td>
<td>186</td>
<td>273</td>
<td>3/15</td>
<td>2-21</td>
<td>1/3/15</td>
</tr>
<tr>
<td>pigs</td>
<td>441</td>
<td>592</td>
<td>2/39</td>
<td>3-3</td>
<td>1/2/41</td>
</tr>
<tr>
<td>link</td>
<td>724</td>
<td>1125</td>
<td>3/14</td>
<td>2-4</td>
<td>0/2/17</td>
</tr>
</tbody>
</table>

Table 4.1. Performance indicators

We first investigate the quality of the skeleton re-
turned by H2PC during the CB phase. To this end, we
measure the false positive edge ratio, the precision (i.e.,
the number of true positive edges in the output divided
by the number of edges in the output), the recall (i.e.,
the number of true positive edges divided the true number
of edges) and a combination of precision and recall de-
finied as \( \sqrt{(1 - \text{precision})^2 + (1 - \text{recall})^2} \), to measure
the Euclidean distance from perfect precision and re-
call, as proposed in (Peña et al., 2007). Second, to as-
se over the 8 benchmark sets, as a measure of how well the network general-
izes to new data;

• the BIC score of the network for a new data set, again as a measure of how well the network gener-
alizes to new data;

• the Structural Hamming Distance (SHD) between
the learned and the true structure of the network,
as a measure of the quality of the learned depen-
dence structure. The SHD between two PDAGs is
defined as the number of the following operators
required to make the PDAGs match: add or delete
an undirected edge, and add, remove, or reverse the
orientation of an edge.

For each data set sampled from the true probability
distribution of the benchmark, we first learn a network
structure with H2PC and MMHC and then we com-
pute the relevant performance indicators for each pair
of network structures. The data set used to assess how
well the network generalizes to new data is generated
again from the true probability structure of the bench-
mark networks and contains 50000 observations.

Notice that using the BDeu score as a metric of recon-
struction quality has the following two problems. First,
the score corresponds to the a posteriori probability of a
network only under certain conditions (e.g., a Dirichlet
distribution of the hyper parameters); it is unknown to
what degree these assumptions hold in distributions en-
countered in practice. Second, the score is highly sensi-
tive to the equivalent sample size (set to 10 in our exper-
iments) and depends on the network priors used. Since,
typically, the same arbitrary value of this parameter is
used both during learning and for scoring the learned
network, the metric favors algorithms that use the BDeu
score for learning. In fact, the BDeu score does not rely
on the structure of the original, gold standard network at
all; instead it employs several assumptions to score the
networks. For those reasons, in addition to the score we
also report the BIC score and the SHD metric.

4.2 Results

In Figure 1, we report the quality of the skeleton ob-
tained with HPC over that obtained with MMPC (be-
fore the SS phase) as a function of the sample size.
Results for each benchmark are not shown here in de-
tail due to space restrictions. For sake of conciseness,
the performance values are averaged over the 8 bench-
marks depicted in Table 1. The increase factor for a
given performance indicator is expressed as the ratio of

3http://www.bnlearn.com/bnrepository
**Figure 1:** Quality of the skeleton obtained with HPC over that obtained with MMPC (after the CB phase). The 2 first figures present mean values aggregated over the 8 benchmarks. The 4 last figures present increase factors of HPC/MMPC, with the median, quartile, and most extreme values (green boxplots), along with the mean value (black line).
the performance value obtained with HPC over that obtained with MMPC (the gold standard). Note that for some indicators, an increase is actually not an improvement but is worse (e.g., false positive rate, Euclidean distance). For clarity, we mention explicitly on the subplots whether an increase factor > 1 should be interpreted as an improvement or not. Regarding the quality of the superstructure, the advantages of HPC against MMPC are noticeable. As observed, HPC consistently increases the recall and reduces the rate of false negative edges. As expected this benefit comes at a little expense in terms of false positive edges. HPC also improves the Euclidean distance from perfect precision and recall on all benchmarks, while increasing the number of independence tests and thus the running time in the CB phase (see number of statistical tests). It is worth noting that HPC is capable of reducing by 50% the Euclidean distance with 50000 samples (lower left plot). These results are very much in line with other experiments presented in (Rodrigues de Morais & Aussem, 2010a; Villanueva & Maciel, 2012).

In Figure 2, we report the quality of the final DAG obtained with H2PC over that obtained with MMHC (after the SS phase) as a function of the sample size. Regarding BDeu and BIC on both training and test data, the improvements are noteworthy. The results in terms of goodness of fit to training data and new data using H2PC clearly dominate those obtained using MMHC, whatever the sample size considered, hence its ability to generalize better. Regarding the quality of the network structure itself (i.e., how close is the DAG to the true dependence structure of the data), this is pretty much a dead heat between the 2 algorithms on small sample sizes (i.e., 50 and 100), however we found H2PC to perform significantly better on larger sample sizes. The SHD increase factor decays rapidly (lower is better) as the sample size increases (lower left plot). For 50 000 samples, the SHD is on average only 50% that of MMHC. Regarding the computational burden involved, we may observe from Table 2 that H2PC has a little computational overhead compared to MMHC. The running time increase factor grows somewhat linearly with the sample size. With 50000 samples, H2PC is approximately 10 times slower on average than MMHC. This is mainly due to the computational expense incurred in obtaining larger PC sets with HPC, compared to MMPC.

5. Application to multi-label learning

In this section, we address the problem of multi-label learning with H2PC. MLC is a challenging problem in many real-world application domains, where each instance can be assigned simultaneously to multiple binary labels (Dembczyński et al., 2012; Read et al., 2009; Madjarov et al., 2012; Kocev et al., 2007; Tsoumakas et al., 2010b). Formally, learning from multi-label examples amounts to finding a mapping from a space of features to a space of labels. We shall assume throughout that \( \mathbf{X} \) (a random vector in \( \mathbb{R}^d \)) is the feature set, \( \mathbf{Y} \) (a random vector in \( \{0, 1\}^n \)) is the label set, \( \mathbf{U} = \mathbf{X} \cup \mathbf{Y} \) and \( p \) a probability distribution defined over \( \mathbf{U} \). Given a multi-label training set \( \mathcal{D} \), the goal of multi-label learning is to find a function which is able to map any unseen example to its proper set of labels. From the Bayesian point of view, this problem amounts to modeling the conditional joint distribution \( p(\mathbf{Y} \mid \mathbf{X}) \).

5.1. Related work

This MLC problem may be tackled in various ways (Luaces et al., 2012; Alvares-Cherman et al., 2011; Read et al., 2009; Blockeel et al., 1998; Kocev et al., 2007). Each of these approaches is supposed to capture - to some extent - the relationships between labels. The two most straightforward meta-learning methods (Madjarov et al., 2012) are: Binary Relevance (BR) (Luaces et al., 2012) and Label Powerset (LP) (Tsoumakas & Vlahavas, 2007; Tsoumakas et al., 2010b). Both methods can be regarded as opposite in the sense that BR does consider each label independently, while LP considers the whole label set at once (one multi-class problem). An important question remains: what shall we capture from the statistical relationships between labels exactly to solve the multi-label classification problem? The problem attracted a great deal of interest (Dembczyński et al., 2012; Zhang & Zhang, 2010). It is well beyond the scope and purpose of this paper to delve deeper into these approaches, we point the reader to (Dembczyński et al., 2012; Tsoumakas et al., 2010b) for a review. The second fundamental problem that we wish to address involves finding an optimal feature subset selection of a label set, w.r.t an Information Theory criterion Koller & Sahami (1996). As in the single-label case, multi-label feature selection has been studied recently and has encountered some success (Gu et al., 2011; Spolaò et al., 2013).

5.2. Label powerset decomposition

We shall first introduce the concept of label powerset that will play a pivotal role in the factorization of the conditional distribution \( p(\mathbf{Y} \mid \mathbf{X}) \).

Definition 1. \( Y_{LP} \subseteq Y \) is called a label powerset iff \( Y_{LP} \perp \mathbf{Y} \setminus Y_{LP} \mid X \). Additionally, \( Y_{LP} \) is said minimal
Figure 2: Quality of the final DAG obtained with H2PC over that obtained with MMHC (after the SS phase). The 6 figures present increase factors of H2PC/MMPC, with the median, quartile, and most extreme values (green boxplots), along with the mean value (black line).
Table 2: Total running time ratio $R$ (H2PC/MMHC). White cells indicate a ratio $R < 1$ (in favor of H2PC), while shaded cells indicate a ratio $R > 1$ (in favor of MMHC). The darker, the larger the ratio.

<table>
<thead>
<tr>
<th>Network</th>
<th>Sample Size</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>500</th>
<th>1000</th>
<th>2000</th>
<th>5000</th>
<th>10000</th>
<th>20000</th>
<th>50000</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td></td>
<td>0.94</td>
<td>0.87</td>
<td>1.14</td>
<td>1.99</td>
<td>2.26</td>
<td>2.12</td>
<td>2.36</td>
<td>2.58</td>
<td>1.75</td>
<td>1.78</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.4</td>
<td>±0.3</td>
<td>±0.6</td>
<td>±0.5</td>
</tr>
<tr>
<td>insurance</td>
<td></td>
<td>0.96</td>
<td>1.09</td>
<td>1.56</td>
<td>2.93</td>
<td>3.06</td>
<td>3.48</td>
<td>3.69</td>
<td>4.10</td>
<td>3.76</td>
<td>3.75</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.3</td>
<td>±0.3</td>
<td>±0.3</td>
<td>±0.4</td>
<td>±0.4</td>
<td>±0.6</td>
<td>±0.5</td>
</tr>
<tr>
<td>mildew</td>
<td></td>
<td>0.77</td>
<td>0.80</td>
<td>0.79</td>
<td>0.94</td>
<td>1.01</td>
<td>1.23</td>
<td>1.74</td>
<td>2.14</td>
<td>3.26</td>
<td>6.20</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.3</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.6</td>
<td>±1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>alarm</td>
<td></td>
<td>0.88</td>
<td>1.11</td>
<td>1.75</td>
<td>2.43</td>
<td>2.55</td>
<td>2.71</td>
<td>2.65</td>
<td>2.80</td>
<td>2.49</td>
<td>2.18</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.6</td>
<td>±0.6</td>
<td>±0.6</td>
<td>±0.6</td>
</tr>
<tr>
<td>hallfinder</td>
<td></td>
<td>0.85</td>
<td>0.85</td>
<td>1.40</td>
<td>1.69</td>
<td>1.83</td>
<td>2.06</td>
<td>2.13</td>
<td>2.12</td>
<td>1.95</td>
<td>1.96</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.2</td>
<td>±0.3</td>
<td>±0.6</td>
<td>±0.6</td>
</tr>
<tr>
<td>munin</td>
<td></td>
<td>0.77</td>
<td>0.85</td>
<td>0.93</td>
<td>1.35</td>
<td>2.11</td>
<td>4.30</td>
<td>12.92</td>
<td>23.32</td>
<td>24.95</td>
<td>24.76</td>
</tr>
<tr>
<td></td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.2</td>
<td>±0.7</td>
<td>±2.6</td>
<td>±5.1</td>
<td>±6.7</td>
<td></td>
</tr>
<tr>
<td>pigs</td>
<td></td>
<td>0.80</td>
<td>0.80</td>
<td>4.55</td>
<td>4.71</td>
<td>5.00</td>
<td>5.62</td>
<td>7.63</td>
<td>11.10</td>
<td>14.02</td>
<td>11.74</td>
</tr>
<tr>
<td></td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.1</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.3</td>
<td>±0.6</td>
<td>±1.7</td>
<td>±3.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>link</td>
<td></td>
<td>1.16</td>
<td>1.93</td>
<td>2.76</td>
<td>5.55</td>
<td>7.04</td>
<td>8.19</td>
<td>9.00</td>
<td>13.87</td>
<td>15.32</td>
<td>24.74</td>
</tr>
<tr>
<td></td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.0</td>
<td>±0.1</td>
<td>±0.2</td>
<td>±0.3</td>
<td>±0.4</td>
<td>±2.5</td>
<td>±4.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>all</td>
<td></td>
<td>0.89</td>
<td>1.04</td>
<td>1.86</td>
<td>2.70</td>
<td>3.11</td>
<td>3.71</td>
<td>5.39</td>
<td>7.75</td>
<td>8.44</td>
<td>9.64</td>
</tr>
<tr>
<td></td>
<td>±0.1</td>
<td>±0.1</td>
<td>±1.2</td>
<td>±1.5</td>
<td>±1.9</td>
<td>±2.2</td>
<td>±4.0</td>
<td>±7.3</td>
<td>±8.4</td>
<td>±9.7</td>
<td></td>
</tr>
</tbody>
</table>

If it is non empty and has no label powerset as proper subset.

Let $\mathbf{LP}$ denote the set of all powersets defined over $\mathbf{U}$, and $\min \mathbf{LP}$ the set of all minimal label powersets. It is easily shown $\{\mathbf{Y}, \emptyset\} \subseteq \mathbf{LP}$. The key idea behind label powersets is the decomposition of the conditional distribution of the labels into a product of factors

$$p(\mathbf{Y} \mid \mathbf{X}) = \prod_{j=1}^{L} p(\mathbf{Y}_{LP_{j}} \mid \mathbf{X}) = \prod_{j=1}^{L} p(\mathbf{Y}_{LP_{j}} \mid \mathbf{M}_{LP_{j}})$$

where $\{\mathbf{Y}_{LP_{1}}, \ldots, \mathbf{Y}_{LP_{L}}\}$ is a partition of label powersets and $\mathbf{M}_{LP_{j}}$ is a Markov blanket of $\mathbf{Y}_{LP_{j}}$ in $\mathbf{X}$. From the above definition, we have $\mathbf{Y}_{LP_{j}} \perp \mathbf{Y}_{LP_{j}} \mid \mathbf{X}, \forall i \neq j$.

In the framework of MLC, one can consider a multitude of loss functions. In this study, we focus on a non label-wise decomposable loss function called the submodular loss functions. In this study, we focus on a non label-wise decomposable loss function called the submodular loss functions. Let $\mathbf{LP}$ denote the set of all powersets defined over $\mathbf{U}$, and $\min \mathbf{LP}$ the set of all minimal label powersets. It is easily shown $\{\mathbf{Y}, \emptyset\} \subseteq \mathbf{LP}$. The key idea behind label powersets is the decomposition of the conditional distribution of the labels into a product of factors

$$p(\mathbf{Y} \mid \mathbf{X}) = \prod_{j=1}^{L} p(\mathbf{Y}_{LP_{j}} \mid \mathbf{X}) = \prod_{j=1}^{L} p(\mathbf{Y}_{LP_{j}} \mid \mathbf{M}_{LP_{j}})$$

where $\{\mathbf{Y}_{LP_{1}}, \ldots, \mathbf{Y}_{LP_{L}}\}$ is a partition of label powersets and $\mathbf{M}_{LP_{j}}$ is a Markov blanket of $\mathbf{Y}_{LP_{j}}$ in $\mathbf{X}$. From the above definition, we have $\mathbf{Y}_{LP_{j}} \perp \mathbf{Y}_{LP_{j}} \mid \mathbf{X}, \forall i \neq j$.

The next section aims to obtain theoretical results for the characterization of the minimal label powersets $\mathbf{Y}_{LP_{j}}$ and their Markov boundaries $\mathbf{M}_{LP_{j}}$ from a DAG, in order to be able to estimate the MPE more effectively.

5.3. Label powerset characterization

In this section, we show that minimal label powersets can be depicted graphically when $p$ is faithful to a DAG.

**Theorem 6.** Suppose $p$ is faithful to a DAG $\mathcal{G}$. Then, $Y_{i}$ and $Y_{j}$ belong to the same minimal label powerset if and only if there exists an undirected path in $\mathcal{G}$ between nodes $Y_{i}$ and $Y_{j}$ in $\mathbf{Y}$ such that all intermediate nodes $Z$ are either (i) $Z \in \mathbf{Y}$, or (ii) $Z \in \mathbf{X}$ and $Z$ has two parents in $\mathbf{Y}$ (i.e. a collider of the form $Y_{p} \rightarrow X \leftarrow Y_{q}$).

The proof is given in the Appendix. We shall now address the following questions: What is the Markov boundary in $\mathbf{X}$ of a minimal label powerset? Answering this question for general distributions is not trivial. We establish a useful relation between a label powerset and its Markov boundary in $\mathbf{X}$ when $p$ is faithful to a DAG,
Theorem 7. Suppose \( p \) is faithful to a DAG \( \mathcal{G} \). Let \( Y_{LP} = \{Y_1, Y_2, \ldots, Y_n\} \) be a label powerset. Then, its Markov boundary \( \mathbf{M} \) in \( \mathbf{U} \) is also its Markov boundary in \( \mathbf{X} \), and is given in \( \mathcal{G} \) by \( \mathbf{M} = \bigcup_{j=1}^{n} \left( \mathbf{PC}_{Y_j} \cup \mathbf{SP}_{Y_j} \right) \) \( \setminus \ Y \).

The proof is given in the Appendix.

5.4. Experimental setup

The MLC problem is decomposed into a series of multi-class classification problems, where each multi-class variable encodes one label powerset, with as many classes as the number of possible combinations of labels, or those present in the training data. At this point, it should be noted that the LP method is a special case of this framework since the whole label set is a particular label powerset (not necessarily minimal though). The above procedure can been summarized as follows:

1. Learn the BN local graph \( \mathcal{G} \) around the label set;
2. Read off \( \mathcal{G} \) the minimal label powersets and their respective Markov boundaries;
3. Train an independent multi-class classifier on each minimal LP, with the input space restricted to its Markov boundary in \( \mathcal{G} \);
4. Aggregate the prediction of each classifier to output the most probable explanation, i.e. \( \arg \max_{\mathbf{y}} \ p(\mathbf{y} \mid \mathbf{x}) \).

To assess separately the quality of the minimal label powerset decomposition and the feature subset selection with Markov boundaries, we investigate 4 scenarios:

- **BR** without feature selection (denoted **BR**): a classifier is trained on each single label with all features as input. This is the simplest approach as it does not exploit any label dependency. It serves as a baseline learner for comparison purposes.
- **BR** with feature selection (denoted **BR+MB**): a classifier is trained on each single label with the input space restricted to its Markov boundary in \( \mathcal{G} \). Compared to the previous strategy, we evaluate here the effectiveness of the feature selection task.
- Minimum label powerset method without feature selection (denoted **MLP**): the minimal label powersets are obtained from the DAG. All features are used as inputs.
- Minimum label powerset method with feature selection (denoted **MLP+MB**): the minimal label powersets are obtained from the DAG, the input space is restricted to the Markov boundary of the labels in that powerset.

We use the same base learner in each meta-learning method: the well-known Random Forest classifier (Breiman, 2001). RF achieves good performance in standard classification as well as in multi-label problems (Kocev et al., 2007; Madjarov et al., 2012), and is able to handle both continuous and discrete data easily, which is much appreciated. The standard RF implementation in \( \text{R} \) (Liaw & Wiener, 2002)\(^4\) was used. For practical purposes, we restricted the forest size of RF to 100 trees, and left the other parameters to their default values.

5.4.1. Data and performance indicators

A total of 10 multi-label data sets are collected for experiments in this section, whose characteristics are summarized in Table 3. These data sets come from different problem domains including text, biology, and music. They can be found on the Mulan\(^5\) repository, except for **image** which comes from Zhou\(^6\) (Maron & Ratan, 1998). Let \( D \) be the multi-label data set, we use \( |D|, \text{dim}(D), L(D), F(D) \) to represent the number of examples, number of features, number of possible labels, and feature type respectively. \( DL(D) = |\{Y \mid \exists x : (x, Y) \in D\}| \) counts the number of distinct label combinations appearing in the data set. Continues values are binarized during the BN structure learning phase.

| data set | domain | \( |D| \) | \text{dim}(D) | \( L(D) \) | \( F(D) \) | \( DL(D) \) |
|----------|---------|-----------|-------------|------------|-----------|-----------|
| emotions | music   | 593       | 72 cont.    | 6          | 27        |
| yeast    | biology | 2417      | 103 cont.   | 14         | 198       |
| image    | images  | 2000      | 135 cont.   | 5          | 20        |
| scene    | images  | 2407      | 294 cont.   | 6          | 15        |
| slashdot | text    | 3782      | 1079 disc.  | 22         | 156       |
| genbase  | biology | 662       | 1186 disc.  | 27         | 32        |
| medical  | text    | 978       | 1449 disc.  | 45         | 94        |
| enron    | text    | 1702      | 1001 disc.  | 53         | 753       |
| bibtex   | text    | 7395      | 1836 disc.  | 159        | 2856      |
| core5k   | images  | 5000      | 499 disc.   | 374        | 3175      |

The performance of a multi-label classifier can be assessed by several evaluation measures (Tsoumakas et al., 2010a). We focus on maximizing a non-decomposable score function: the global accuracy (also termed subset accuracy, complement of the 0/1-loss), which measures the correct classification rate of the whole label set (exact match of all labels required).

\(^4\)http://cran.r-project.org/web/packages/randomForest

\(^5\)http://mulan.sourceforge.net/datasets.html

\(^6\)http://lamda.nju.edu.cn/data_MIMLImage.ashx
Note that the global accuracy implicitly takes into account the label correlations. It is therefore a very strict evaluation measure as it requires an exact match of the predicted and the true set of labels. It was recently proved in (Dembczynski et al., 2012) that BR is optimal for decomposable loss functions (e.g., the hamming loss), while non-decomposable loss functions (e.g. subset loss) inherently require the knowledge of the label conditional distribution. 10-fold cross-validation was performed for the evaluation of the MLC methods.

5.4.2. Results

Table 4 reports the outputs of H2PC and MMHC. Table 5 shows the global accuracy of each method on the 10 data sets. Table 6 reports the running time and the average degree of the label nodes in Table 6 that several inents. First, we may observe by inspection of the aver-

In contrast, on genbase the MLP approach boils down to the simple BR scheme. This is confirmed by inspect-

ing Table 5: the performances of MMHC and H2PC (without feature selection) are equal. An interesting ob-

ervation upon looking at the distribution of the label powerset size shows that for the remaining data sets, the MLP mostly decomposes the label set in two parts: one on which it performs BR and the other one on which it performs LP. Take for instance enron, it can be seen from Table 4 that there are approximately 23 label sin-
gletons and a powerset of 30 labels with H2PC for a total of 53 labels. The gain in performance with MLP over BR, our baseline learner, can be ascribed to the quality of the label powerset decomposition as BR is ignoring label dependencies. As expected, the results using MLP clearly dominate those obtained using BR on all data sets except genbase. The DAGs display the minimum label powersets and their relevant features.

Third, H2PC compares favorably to MMHC. On scene for instance, the accuracy of MLP+MB has raised from 20% (with MMHC) to 56% (with H2PC). On yeast, it has raised from 7% to 23%. Without feature selection, the difference in global accuracy is less pronounced but still in favor of H2PC. A Wilcoxon signed rank paired test reveals statistically significant improvements of H2PC over MMHC in the MLP approach without feature selection ($p < 0.02$). This trend is more pronounced when the feature selection is used ($p < 0.001$), using MLP-MB. Note however that the LP decomposi-
tion for H2PC and MMHC on yeast and image are strictly identical, hence the same accuracy values in Ta-

ble 5 in the MLP column.

Fourth, regarding the utility of the feature selection, it is difficult to reach any conclusion. Whether BR or MLP is used, the use of the selected features as inputs to the classification model is not shown greatly beneficial in terms of global accuracy on average. The performance of BR and our MLP with all the features outperforms that with the selected features in 6 data sets but the feature selection leads to actual improvements in 3 data sets. The difference in accuracy with and with-

out the feature selection was not shown to be statistically significant ($p > 0.20$ with a Wilcoxon signed rank paired test). Surprisingly, the feature selection did exceptionally well on genbase. On this data set, the in-

crease in accuracy is the most impressive: it raised from 7% to 98% which is atypical. The dramatic increase in accuracy on genbase is due solely to the restricted feature set as input to the classification model. This is also observed on medical, to a lesser extent though. Interest-

ingly, on large and densely connected networks (e.g. bibtext, slashdot and corel5K), the feature selection per-
formed very well in terms of global accuracy and significantly reduced the input space which is noticeable. On emotions, yeast, image, scene and genbase, the method reduced the feature set down to nearly 1/100 its original size. The feature selection should be evaluated in view of its effectiveness at balancing the increasing error and the decreasing computational burden by drastically reducing the feature space. We should also keep in mind that the feature relevance cannot be defined independently of the learner and the model-performance metric (e.g., the loss function used). Admittedly, our feature selection based on the Markov boundaries is not necessarily optimal for the base MLC learner used here, namely the Random Forest model.

As far as the overall running time performance is concerned, we see from Table 6 that for both methods, the running time grows somewhat exponentially with the parent set size of the true underlying network. The running time is known to increase exponentially that of MMHC (which may hurt interpretability). In contrast, constraint-based heuristics (Perrier et al., 2008; Kojima et al., 2010) in terms of goodness of fit and structure effectiveness at balancing the increasing error and the decreasing computational burden by drastically reducing the feature space. We should also keep in mind that the feature relevance cannot be defined independently of the learner and the model-performance metric (e.g., the loss function used). Admittedly, our feature selection based on the Markov boundaries is not necessarily optimal for the base MLC learner used here, namely the Random Forest model.

6. Discussion & practical applications

Our prime conclusion is that H2PC is a promising approach to constructing BN global or local structures around specific nodes of interest, with potentially thousands of variables. Concentrating on higher recall values while keeping the false positive rate as low as possible pays off in terms of computational gains as we discuss in the Conclusion.

### Table 4: Distribution of the number and the size of the minimal label powersets output by H2PC (top) and MMHC (bottom). On each data set: number of minimal powersets, minimum/median/maximum number of labels per powerset, and minimum/median/maximum number of distinct labels per powerset. The total number of labels and distinct labels combinations is recalled for convenience.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># LPs</th>
<th># labels / LP</th>
<th>L(D)</th>
<th># classes / LP</th>
<th>DL(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>emotions</td>
<td>1.0 ± 0.0</td>
<td>6 / 6 / 6</td>
<td>(6)</td>
<td>26 / 27 / 27</td>
<td>(27)</td>
</tr>
<tr>
<td>yeast</td>
<td>2.0 ± 0.0</td>
<td>2 / 7 / 12</td>
<td>(14)</td>
<td>3 / 64 / 135</td>
<td>(198)</td>
</tr>
<tr>
<td>image</td>
<td>1.0 ± 0.0</td>
<td>5 / 5 / 5</td>
<td>(5)</td>
<td>19 / 20 / 20</td>
<td>(20)</td>
</tr>
<tr>
<td>scene</td>
<td>1.0 ± 0.0</td>
<td>6 / 6 / 6</td>
<td>(6)</td>
<td>14 / 15 / 15</td>
<td>(15)</td>
</tr>
<tr>
<td>slashdot</td>
<td>9.5 ± 0.8</td>
<td>1 / 1 / 14</td>
<td>(22)</td>
<td>1 / 2 / 111</td>
<td>(156)</td>
</tr>
<tr>
<td>genbase</td>
<td>26.9 ± 0.3</td>
<td>1 / 1 / 2</td>
<td>(27)</td>
<td>1 / 2 / 2</td>
<td>(32)</td>
</tr>
<tr>
<td>medical</td>
<td>38.6 ± 1.8</td>
<td>1 / 1 / 6</td>
<td>(45)</td>
<td>1 / 2 / 10</td>
<td>(94)</td>
</tr>
<tr>
<td>enron</td>
<td>24.5 ± 1.9</td>
<td>1 / 1 / 30</td>
<td>(53)</td>
<td>1 / 2 / 334</td>
<td>(753)</td>
</tr>
<tr>
<td>bibtex</td>
<td>39.6 ± 4.3</td>
<td>1 / 1 / 112</td>
<td>(159)</td>
<td>2 / 2 / 1588</td>
<td>(2856)</td>
</tr>
<tr>
<td>corel5k</td>
<td>97.7 ± 3.9</td>
<td>1 / 1 / 263</td>
<td>(374)</td>
<td>1 / 2 / 2749</td>
<td>(3175)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th># LPs</th>
<th># labels / LP</th>
<th>L(D)</th>
<th># classes / LP</th>
<th>DL(D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>emotions</td>
<td>1.2 ± 0.4</td>
<td>1 / 6 / 6</td>
<td>(6)</td>
<td>2 / 26 / 27</td>
<td>(27)</td>
</tr>
<tr>
<td>yeast</td>
<td>2.0 ± 0.0</td>
<td>1 / 7 / 13</td>
<td>(14)</td>
<td>2 / 89 / 136</td>
<td>(198)</td>
</tr>
<tr>
<td>image</td>
<td>1.0 ± 0.0</td>
<td>5 / 5 / 5</td>
<td>(5)</td>
<td>19 / 20 / 20</td>
<td>(20)</td>
</tr>
<tr>
<td>scene</td>
<td>1.0 ± 0.0</td>
<td>6 / 6 / 6</td>
<td>(6)</td>
<td>14 / 15 / 15</td>
<td>(15)</td>
</tr>
<tr>
<td>slashdot</td>
<td>15.1 ± 0.6</td>
<td>1 / 1 / 9</td>
<td>(22)</td>
<td>1 / 2 / 46</td>
<td>(156)</td>
</tr>
<tr>
<td>genbase</td>
<td>27.0 ± 0.0</td>
<td>1 / 1 / 11</td>
<td>(27)</td>
<td>1 / 2 / 2</td>
<td>(32)</td>
</tr>
<tr>
<td>medical</td>
<td>41.7 ± 1.4</td>
<td>1 / 1 / 4</td>
<td>(45)</td>
<td>1 / 2 / 7</td>
<td>(94)</td>
</tr>
<tr>
<td>enron</td>
<td>36.8 ± 2.7</td>
<td>1 / 1 / 12</td>
<td>(53)</td>
<td>1 / 2 / 119</td>
<td>(753)</td>
</tr>
<tr>
<td>bibtex</td>
<td>77.2 ± 3.1</td>
<td>1 / 1 / 28</td>
<td>(159)</td>
<td>2 / 2 / 233</td>
<td>(2856)</td>
</tr>
<tr>
<td>corel5k</td>
<td>165.3 ± 5.5</td>
<td>1 / 1 / 177</td>
<td>(374)</td>
<td>1 / 2 / 2294</td>
<td>(3175)</td>
</tr>
</tbody>
</table>

### Table 5: Global classification accuracies using 4 learning methods (BR, MLP, BR+MB, MLP+MB) based on the DMC obtained with H2PC and MMHC. Best values between H2PC and MMHC are bold-faced.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BR</th>
<th>MLP</th>
<th>BR+MB</th>
<th>MLP+MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>emotions</td>
<td>0.327</td>
<td>0.371</td>
<td>0.391</td>
<td>0.147</td>
</tr>
<tr>
<td>yeast</td>
<td>0.169</td>
<td>0.273</td>
<td>0.271</td>
<td>0.062</td>
</tr>
<tr>
<td>image</td>
<td>0.342</td>
<td>0.504</td>
<td>0.504</td>
<td>0.227</td>
</tr>
<tr>
<td>scene</td>
<td>0.580</td>
<td>0.733</td>
<td>0.733</td>
<td>0.123</td>
</tr>
<tr>
<td>slashdot</td>
<td>0.355</td>
<td>0.419</td>
<td>0.474</td>
<td>0.274</td>
</tr>
<tr>
<td>genbase</td>
<td>0.069</td>
<td>0.069</td>
<td>0.069</td>
<td>0.974</td>
</tr>
<tr>
<td>medical</td>
<td>0.454</td>
<td>0.499</td>
<td>0.502</td>
<td>0.630</td>
</tr>
<tr>
<td>enron</td>
<td>0.134</td>
<td>0.165</td>
<td>0.180</td>
<td>0.024</td>
</tr>
<tr>
<td>bibtex</td>
<td>0.108</td>
<td>0.115</td>
<td>0.167</td>
<td>0.120</td>
</tr>
<tr>
<td>corel5k</td>
<td>0.002</td>
<td>0.030</td>
<td>0.043</td>
<td>0.000</td>
</tr>
</tbody>
</table>
The advantages in terms of structure accuracy and its ability to scale to thousands of variables opens up many avenues of future possible applications of H2PC in various domains as we shall discuss next. For example, BNs have especially proven to be useful abstractions in computational biology (Nagarajan et al., 2013; Scutari & Nagarajan, 2013; Prestat et al., 2013; Aussem et al., 2012, 2010; Peña, 2008; Peña et al., 2005). Identifying the gene network is crucial for understanding the behavior of the cell which, in turn, can lead to better diagnosis and treatment of diseases. This is also of great importance for characterizing the function of genes and the proteins they encode in determining traits, psychology, or development of an organism. Genome sequencing uses high-throughput techniques like DNA microarrays, proteomics, metabolomics and mutation analysis to describe the function and interactions of thousands of genes (Zhang & Zhou, 2006). Learning BN models of gene networks from these huge data is still a difficult task (Badea, 2004; Bernard & Hartemink, 2005; Friedman et al., 1999a; Ott et al., 2004; Peer et al., 2001). In these studies, the authors had decide in advance which genes were included in the learning process, in all the cases less than 1000, and which genes were excluded from it (Peña, 2008). H2PC overcome the problem by focusing the search around a targeted gene: the key step is the identification of the vicinity of a node X (Peña et al., 2005).

Our second objective in this study was to demonstrate the potential utility of hybrid BN structure discovery to multi-label learning. In multi-label data where many inter-dependencies between the labels may be present, explicitly modeling all relationships between the labels is intuitively far more reasonable (as demonstrated in our experiments). BNs explicitly account for such inter-dependencies and the DAG allows us to identify an optimal set of predictors for every label powerset. The experiments presented here support the conclusion that local structural learning in the form of local neighborhood induction and Markov blanket is a theoretically well-motivated approach that can serve as a powerful learning framework for label dependency analysis geared toward multi-label learning. Multi-label scenarios are found in many application domains, such as multimedia annotation (Snoek et al., 2006; Trohidis et al., 2008), tag recommendation, text categorization (McCallum, 1999; Zhang & Zhou, 2006), protein function classification (Roth & Fischer, 2007), and antiretroviral drug categorization (Borchani et al., 2013).

### 7. Conclusion & avenues for future research

We first discussed a hybrid algorithm for global or local (around target nodes) BN structure learning called Hybrid HPC (H2PC). Our extensive experiments showed that H2PC outperforms the state-of-the-art MMHC by a significant margin in terms of edge recall without sacrificing the number of extra edges, which is crucial for the soundness of the super-structure used during the second stage of hybrid methods, like the ones proposed in (Perrier et al., 2008; Kojima et al., 2010). The code of H2PC is open-source and publicly available online at https://github.com/madbix/bnlearn-clone-3.4. Second, we discussed an application of H2PC to the multi-label learning problem which is a challenging problem in many real-world application domains. We established theoretical results, under the faithfulness condition, in order to characterize graphically the so-called minimal label powersets that appear as irreducible factors in the joint distribution and their respective Markov boundaries. As far as we know, this is the first investigation of Markov boundary principles to the optimal variable/feature selection problem in multi-label learning. These formal results offer a simple guideline to characterize graphically: i) the minimal label powerset decomposition, (i.e. into minimal subsets \( Y_{LP} \subseteq \mathbb{Y} \) such that \( Y_{LP} \perp \mathbb{Y} \setminus Y_{LP} \mid X \), and ii) the minimal subset of features, w.r.t an Information Theory criterion, needed to predict each label powerset, thereby reducing the input space and the computational burden of the multi-label classification. The theoretical analysis laid the foundation for a practical multi-label classification procedure. Another set of experiments were carried...
out on a broad range of multi-label data sets from different problem domains to demonstrate its effectiveness. H2PC was shown to outperform MMHC by a significant margin in terms of global accuracy.

We suggest several avenues for future research. As far as BN structure learning is concerned, future work will aim at: 1) ascertaining which independence test (e.g. tests targeting specific distributions, employing parametric assumptions etc.) is most suited to the data at hand (Tsamardinos & Borboudakis, 2010; Scutari, 2011); 2) controlling the false discovery rate of the edges in the graph output by H2PC (Peña, 2008) especially when dealing with more nodes than samples, e.g. learning gene networks from gene expression data. In this study, H2PC was run independently on each node without keeping track of the dependencies found previously. This lead to some loss of efficiency due to redundant calculations. The optimization of the H2PC code is currently being undertaken to lower the computational cost while maintaining its performance. These optimizations will include the use of a cache to store the (in)dependencies and the use of a global structure. Other interesting research avenues to explore are extensions and modifications to the greedy search-and-score procedure which is the most promising part to optimize in terms of computational gains. Regarding the multi-label learning problem, experiments with several thousands of labels are currently been conducted and will be reported in due course. We also intend to work on relaxing the faithfulness assumption and derive a practical multi-label classification algorithm based on H2PC that is correct under milder assumptions underlying the joint distribution (e.g. Composition, Intersection). This needs further substantiation through more analysis.

Acknowledgments

The authors thank Marco Scutari for sharing his bnlearn package in R. The research was also supported by grants from the European ENIAC Joint Undertaking (INTEGRATE project) and from the French Rhône-Alpes Complex Systems Institute (IXXI).

Appendix

**Lemma 8.** \( \forall Y_{LP}, Y_{LP}' \subseteq L P \), then \( Y_{LP} \cap Y_{LP}' \subseteq \text{LP} \).

**Proof.** To keep the notation uncluttered and for the sake of simplicity, consider a partition \( \{Y_1, Y_2, Y_3, Y_4\} \) of \( Y \) such that \( Y_{LP} = Y_1 \cup Y_2 \), \( Y_{LP}' = Y_2 \cup Y_3 \). From the label powerset assumption for \( Y_{LP} \) and \( Y_{LP}' \), we have \( (Y_1 \cup Y_2) \perp (Y_3 \cup Y_4) \mid X \) and \( (Y_2 \cup Y_3) \perp (Y_1 \cup Y_4) \mid X \). From the Decomposition, we have that \( Y_2 \perp Y_1 \mid (Y_3 \cup Y_4 \cup X) \). Using the Weak union, we obtain that \( Y_2 \perp Y_1 \mid (Y_3 \cup Y_4 \cup X) \). From these two facts, we can use the Contraction property to show that \( Y_2 \perp Y_1 \mid (Y_1 \cup Y_3 \cup Y_4) \mid X \). Therefore, \( Y_2 = Y_{LP} \cap Y_{LP}' \) is a label powerset by definition.

**Lemma 9.** Let \( Y_i \) and \( Y_j \) denote two distinct labels in \( Y \) and define by \( Y_{LP} \) and \( Y_{LP}' \) their respective minimal label powerset. Then we have:

\[ \exists Z \subseteq Y \setminus \{Y_i, Y_j\}, \{Y_i\} \perp \{Y_j\} \mid (X \cup Z) \Rightarrow Y_{LP} = Y_{LP}' \]

**Proof.** Let us suppose \( Y_{LP} \neq Y_{LP}' \). By the label powerset definition for \( Y_{LP} \), we have \( Y_{LP} \perp Y \setminus Y_{LP} \mid X \). As \( Y_{LP} \cap Y_{LP}' = \emptyset \) owing to Lemma 8, we have that \( Y_j \in Y_{LP}' \Rightarrow \forall Z \subseteq Y \setminus \{Y_i, Y_j\}, Z \) can be decomposed as \( Z = Z_i \cup Z_j \) such that \( Z_i = Z \cap (Y_{LP} \setminus \{Y_j\}) \) and \( Z_j = Z \setminus Z_i \). Using the Decomposition property, we have that \( (\{Y_i\} \cup Z_i) \perp (\{Y_j\} \cup Z_j) \mid X \). Using the Weak union property, we have that \( \{Y_i\} \perp \{Y_j\} \mid (X \cup Z_j) \). As this is true \( \forall Z \subseteq Y \setminus \{Y_i, Y_j\} \), then \( \exists Z \subseteq Y \setminus \{Y_i, Y_j\}, \{Y_i\} /\perp \{Y_j\} \mid (X \cup Z) \), which completes the proof.

**Lemma 10.** Consider \( Y_{LP} \) a minimal label powerset. Then, if \( p \) satisfies the Composition property, \( Z \perp Y_{LP} \setminus Z \mid X \).

**Proof.** By contradiction, suppose a nonempty \( Z \) exists, such that \( Z \perp Y_{LP} \setminus Z \mid X \). From the label powerset assumption of \( Y_{LP} \), we have that \( Y_{LP} \perp Y \setminus Y_{LP} \mid X \). From these two facts, we can use the Composition property to show that \( Z \perp Y \setminus Z \mid X \) which contradicts the minimal label powerset assumption of \( Y_{LP} \). This concludes the proof.

**Theorem 7.** Suppose \( p \) is faithful to a DAG \( G \). Then, \( Y_i \) and \( Y_j \) belong to the same minimal label powerset if and only if there exists an undirected path in \( G \) between nodes \( Y_i \) and \( Y_j \) in \( Y \) such that all intermediate nodes \( Z \) are either (i) \( Z \in Y \), or (ii) \( Z \in X \) and \( Z \) has two parents in \( Y \) (i.e. a collider of the form \( Y_p \rightarrow X \leftarrow Y_q \)).

**Proof.** Suppose such a path exists. By conditioning on all the intermediate colliders \( Y_i \) in \( Y \) of the form \( Y_p \rightarrow Y_i \leftarrow Y_q \) along an undirected path in \( G \) between nodes \( Y_i \) and \( Y_j \), we ensure that \( \exists Z \subseteq Y \setminus \{Y_i, Y_j\}, \text{dSep}(Y_i, Y_j) \mid X \cup Z \). Due to the faithfulness, this is equivalent to \( \{Y_i\} \perp \{Y_j\} \mid (X \cup Z) \). From Lemma 9, we conclude that \( Y_i \) and \( Y_j \) belong to the same minimal label powerset.
To show the inverse, note that owing to Lemma 10, we know that there exists no partition \([Z_i, Z_j]\) of \(Z\) such that \(((Y_i) \cup Z_i) \perp\!\!\!\perp ((Y_j) \cup Z_j) \mid X\). Due to the faithfulness, there exists at least a link between \(((Y_i) \cup Z_i)\) and \(((Y_j) \cup Z_j)\) in the DAG, hence by recursion, there exists a path between \(Y_i\) and \(Y_j\) such that all intermediate nodes \(Z\) are either (i) \(Z \in Y\), or (ii) \(Z \in X\) and \(Z\) has two parents in \(Y\) (i.e. a collider of the form \(Y_p \rightarrow X \leftarrow Y_q\)).

**Theorem 8.** Suppose \(p\) is faithful to a DAG \(G\). Let \(Y_{LP} = \{Y_1, Y_2, \ldots, Y_n\}\) be a label powerset. Then, its Markov boundary \(M\) in \(U\) is also its Markov boundary in \(X\), and is given in \(G\) by \(M = \bigcup_{i=1}^{n} [PC_{Y_i} \cup SP_{Y_i}] \setminus Y\).

Proof. First, we prove that \(M\) is a Markov boundary of \(Y_{LP}\) in \(U\). Define \(M_i\), the Markov boundary of \(Y_i\) in \(U\), and \(M'_i = M_i \setminus Y\). From Theorem 5, \(M_i\) is given in \(G\) by \(M_i = PC_{Y_i} \cup SP_{Y_i}\). We may now prove that \(M'_1 \cup \cdots \cup M'_n\) is a Markov boundary of \(\{Y_1, Y_2, \ldots, Y_n\}\) in \(U\). We show first that the statement holds for \(n = 2\) and then conclude that it holds for all \(n\) by induction.

Let \(W\) denote \(U \setminus (Y_1 \cup Y_2 \cup M'_1 \cup M'_2)\), and define \(Y_1 = \{Y_1\}\) and \(Y_2 = \{Y_2\}\). From the Markov blanket assumption for \(Y_1\) we have \(Y_1 \perp\!\!\!\perp U \setminus (Y_1 \cup M_1) \mid M_1\). Using the Weak Union property, we obtain that \(Y_1 \perp\!\!\!\perp W \mid M'_1 \cup M'_2 \cup Y_2\). Similarly we can derive \(Y_2 \perp\!\!\!\perp W \mid M'_1 \cup M'_2 \cup Y_1\). Combining these two statements yields \(Y_1 \cup Y_2 \perp\!\!\!\perp W \mid M'_1 \cup M'_2\) due to the Intersection property. Let \(M = M'_1 \cup M'_2\) the last expression can be formulated as \((Y_1 \cup Y_2) \perp\!\!\!\perp U \setminus (Y_1 \cup Y_2 \cup M) \mid M\) which is the definition of a Markov blanket of \(Y_1 \cup Y_2\) in \(U\).

We shall now prove that \(M\) is minimal. Let us suppose that it is not the case, i.e., \(\exists Z \subseteq M\) such that \(Y_1 \cup Y_2 \perp\!\!\!\perp Z \cup (U \setminus (Y_1 \cup Y_2 \cup M)) \mid M\). Define \(Z_1 = Z \cap M_1\) and \(Z_2 = Z \cap M_2\). We may apply the Weak Union property to get \(Y_1 \perp\!\!\!\perp Z_1 \mid (M \setminus Z) \cup Y_2 \cup (U \setminus (Y_1 \cup M))\) which can be rewritten more compactly as \(Y_1 \perp\!\!\!\perp Z_1 \mid (M_1 \setminus Z_1) \cup (U \setminus (Y_1 \cup M))\). From the Markov blanket assumption, we have \(Y_1 \perp\!\!\!\perp U \setminus (Y_1 \cup M_1) \mid M_1\). We may now apply the Intersection property on these two statements to obtain \(Y_1 \perp\!\!\!\perp Z_1 \cup (U \setminus (Y_1 \cup M_1)) \mid M_1 \setminus Z_1\). Similarly, we can derive \(Y_2 \perp\!\!\!\perp Z_2 \cup (U \setminus (Y_2 \cup M_2)) \mid M_2 \setminus Z_2\). From the Markov boundary assumption of \(M_1\) and \(M_2\), we have necessarily \(Z_1 = \emptyset\) and \(Z_2 = \emptyset\), which in turn yields \(Z = \emptyset\). To conclude for any \(n > 2\), it suffices to set \(Y_1 = \bigcup_{i=1}^{n} (Y_i)\) and \(Y_2 = \{Y_1\}\) to conclude by induction.

Second, we prove that \(M\) is a Markov blanket of \(Y_{LP}\) in \(X\). Define \(Z = M \cap Y\). From the label powerset definition, we have \(Y_{LP} \perp\!\!\!\perp Y \mid Y_{LP} \setminus X\). Using the Weak Union property, we obtain \(Y_{LP} \perp\!\!\!\perp Z \mid U \setminus (Y_{LP} \cup Z)\) which can be reformulated as \(Y_{LP} \perp\!\!\!\perp Z \mid (U \setminus (Y_{LP} \cup M_1)) \cup (M \setminus Z)\). Now, the Markov blanket assumption for \(Y_{LP}\) in \(U\) yields \(Y_{LP} \perp\!\!\!\perp U \setminus (Y_{LP} \cup M) \mid M\) which can be rewritten as \(Y_{LP} \perp\!\!\!\perp U \setminus (Y_{LP} \cup M) \mid (M \setminus Z) \cup Z\). From the Intersection property, we get \(Y_{LP} \perp\!\!\!\perp Z \cup U \setminus (Y_{LP} \cup M) \mid M \setminus Z\). From the Markov boundary assumption of \(M\) in \(U\), we know that there exists no proper subset of \(M\) which satisfies this statement, and therefore \(Z = M \cap Y = \emptyset\). From the Markov blanket assumption of \(M\) in \(U\), we have \(Y_{LP} \perp\!\!\!\perp U \setminus (Y_{LP} \cup M) \mid M\). Using the Decomposition property, we obtain \(Y_{LP} \perp\!\!\!\perp X \setminus M \mid M\) which, together with the assumption \(M \cap Y = \emptyset\), is the definition of a Markov blanket of \(Y_{LP}\) in \(X\).

Finally, we prove that \(M\) is a Markov boundary of \(Y_{LP}\) in \(X\). Let us suppose that it is not the case, i.e., \(\exists Z \subseteq M\) such that \(Y_{LP} \perp\!\!\!\perp Z \cup (X \setminus M) \mid M \setminus Z\). From the label powerset assumption of \(Y_{LP}\), we have \(Y_{LP} \perp\!\!\!\perp Y \setminus Y_{LP} \mid X\) which can be rewritten as \(Y_{LP} \perp\!\!\!\perp Y \setminus Y_{LP} \mid (X \setminus M) \cup (M \setminus Z) \cup Z\). Due to the Contraction property, combining these two statements yields \(Y_{LP} \perp\!\!\!\perp Z \cup (U \setminus (M \cup Y_{LP})) \mid M \setminus Z\). From the Markov boundary assumption of \(M\) in \(U\), we necessarily \(Z = \emptyset\), which suffices to prove that \(M\) is a Markov boundary of \(Y_{LP}\) in \(X\).
Figure 3: The local BN structures learned by MMHC (left plot) and H2PC (right plot) on a single cross-validation split, on Emotions and Yeast.
Figure 4: The local BN structures learned by MMHC (left plot) and H2PC (right plot) on a single cross-validation split, on Image and Scene.
Figure 5: The local BN structures learned by MMHC (left plot) and H2PC (right plot) on a single cross-validation split, on Slashdot and Genbase.
Figure 6: The local BN structures learned by MMHC (left plot) and H2PC (right plot) on a single cross-validation split, on Medical and Enron.
Figure 7: The local BN structures learned by MMHC (left plot) and H2PC (right plot) on a single cross-validation split, on Bibtex and Corel5k.
References


