Learning Markov networks with context-specific independences

Alejandro Edera, Federico Schlüter, Facundo Bromberg
Departamento de Sistemas de Información,
Universidad Tecnológica Nacional, Facultad Regional Mendoza, Argentina.
{*aedera,federico.schluter,fbromberg*}*@frm.utn.edu.ar*

***Abstract*—Learning the Markov network structure from data is a problem that has received considerable attention in machine learning, and in many other application fields. This work focuses on a particular approach for this purpose called *independence-based* learning. Such approach guarantees the learning of the correct structure efficiently, whenever data is sufficient for representing the underlying distribution. However, an important issue of such approach is that the learned structures are encoded in an undirected graph. The problem with graphs is that they cannot encode some types of independence relations, such as the context-specific independences. They are a particular case of conditional independences that is true only for a certain assignment of its conditioning set, in contrast to conditional independences that must hold for all its assignments. In this work we present CSPC, an independence-based algorithm for learning structures that encode context-specific independences, and encoding them in a log-linear model, instead of a graph. The central idea of CSPC is combining the theoretical guarantees provided by the independence-based approach with the benefits of representing complex structures by using features in a log-linear model. We present experiments in a synthetic case, showing that CSPC is more accurate than the state-of-the-art IB algorithms when the underlying distribution contains CSIs.**

***Keywords*-Markov networks, structure learning; independence-based; context-specific independences;**

I. Introduction

Nowadays, a powerful representation of joint probability distributions are Markov networks. The structure of a Markov network can encode complex probabilistic relationships among the variables of the domain, improving the efficiency in the procedures for probabilistic inference. An important problem is learning the structure from samples drawn from an unknown distribution. A number of alternative algorithms for this purpose have been developed in recent years. One approach is the *independence-based* (IB) approach [1]–[5]. Algorithms that follow this approach proceed by using statistical tests to learn a series of conditional independences from data, encoding them in an undirected graph. An important advantage of this approach is that it provides theoretical guarantees for learning the correct structure, together with the efficiency gained by using statistical tests. Other recent approaches [6]–[9] proceed by inducing a set of features from data, instead of an undirected graph. The features are real-valued functions of partial variable assignments, and using these functions it is possible to encode more complex structures than those encoded by graphs. Algorithms that follow this approach encode the structure in the features of a log-linear model. Unfortunately, current algorithms based on learning features are not an efficient alternative, due to the multiple user defined hyper-parameters, and the need of performing parameters learning. The parameters learning step is often intractable, requiring an iterative optimization that runs an inference step over the model at each iteration.

In many practical cases the underlying distribution of a problem present *context-specific independences* (CSIs) [10], that are conditional independences that only hold for a certain assignment of the conditioning set, but not hold for the remaining assignments. In that case, encoding the structure in an undirected graph leads to excessively dense graphs, obscuring the CSIs present in the distribution, and resulting in computationally more expensive computation of inference algorithms [11], [12]. For this reason, encoding the CSIs in a log-linear model does not obscure them, achieving sparser models and therefore significant improvements in time, space and sample complexities [8], [13]–[15].

This work presents CSPC, an independence-based algorithm for learning a set of features instead of a graph, in order to encode CSIs. The algorithm is designed as an adaptation for this purpose of the well-known PC algorithm [16]. CSPC proceeds by first generating an initial set of features from the dataset, and then searches over the space of possible contexts for learning the CSIs present in the underlying distribution. For each context the algorithm elicits a set of CSIs using statistical tests, and generalizes the current set of features in order to encode the elicited CSIs. The central idea of CSPC is combining the theoretical guarantees provided by the independence-based approach with the benefits of representing complex structures by using features. To our knowledge, the only algorithm near to CSPC is the LEM algorithm [8], since it uses statistical tests to learn CSIs. However, LEM restricts the attention to learning distributions that can be represented by decomposable Markov networks. For the latter, we omit it as competitor in our experiments.

We conducted an empirical evaluation on synthetic data generated from known distributions that contains CSIs. In our experiments we prove that CSPC is significantly more accurate than the state-of-the-art IB algorithms when the underlying distribution contains CSIs.

II. Background

This section reviews the basics about Markov networks representation, the concept of CSIs, and the IB approach for learning Markov networks.

A. Markov networks

A Markov network over a domain X of n random variables $X_0 \dots X_{n-1}$ is represented by an undirected graph G with n nodes and a set of numerical parameters $\theta \in \mathbb{R}$. This representation can be used to factorize the distribution with the Hammersley-Clifford theorem [17], by using the completely connected sub-graphs of G (a.k.a., *cliques*) into a set of *potential functions* $\{\phi_C(X_C) : C \in cliques(G)\}$ of lower dimension than $p(X)$, parameterized by θ, as follows:

$$p(X = x) = \frac{1}{Z} \prod_{C \in cliques(G)} \phi_C(x_C), \qquad (1)$$

where x is a complete assignment of the domain X, x_C is the projection of the assignment x over the variables of the Cth clique, and Z is a normalization constant. An often used alternative representation is a *log-linear* model, with each clique potential represented as an exponentiated weighted sum of features of the assignment, as follows:

$$p(X = x) = \frac{1}{Z} \exp\left\{ \sum_j \theta_j f_j(x) \right\}, \qquad (2)$$

where each feature f_j is a partial assignment over a subset of the domain $V(f_j)$. Given an assignment x, a feature f_j is said to be satisfied iff for each single variable $X_a = x_a \in f_j$ it also holds that $x_a \in x$ [8]. One can associate a indicator function to f_j and an assignment x by associating a value 1 when f_j is satisfied in x, or 0 otherwise.

A Markov network can be induced from a log-linear model by adding an edge in the graph between every pair of variables X_a, X_b that appear together in some subset of a feature f_j, that is $\{X_a, X_b\} \subseteq V(f_j)$. Then, the clique potentials are constructed from the log-linear features in the obvious way [18].

Example 1. *Figure 1 shows the features of a log-linear model over $n = 3$ binary variables X_f, X_a and X_b, and its respective induced graph.*

B. Context-specific independences

The CSIs are a finer-grained type of independences. These independences are similar to conditional independences, but hold for a specific assignment of the conditioning set, called the *context* of the independence. Formally, we define a CSI as follows:

f_1 $(X_a = 0 \;\; X_b = 0 \;\; X_f = 0)$
f_2 $(X_a = 1 \;\; X_b = 0 \;\; X_f = 0)$
f_3 $(X_a = 0 \;\; X_b = 1 \;\; X_f = 0)$
f_4 $(X_a = 1 \;\; X_b = 1 \;\; X_f = 0)$ $\Rightarrow$
f_5 $(X_a = 0 \; X_f = 1)$
f_6 $(X_a = 1 \; X_f = 1)$
f_7 $(X_b = 0 \; X_f = 1)$
f_8 $(X_b = 1 \; X_f = 1)$

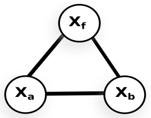

Figure 1: An example of an induced graph from a set of features.

Definition 1 (Context-specific independence [10])**.** *Let $X_a, X_b \in X$ be two random variables, $X_U, X_W \subseteq X \setminus \{X_a, X_b\}$ be pairwise disjoint sets of variables that does not contain X_a, X_b; and x_W some assignment of X_W. We say that variables X_a and X_b are* contextually independent *given X_U and a context $X_W = x_W$, denoted $I(X_a, X_b \mid X_U, x_W)$, iff*

$$p(X_a | X_b, X_U, x_W) = p(X_a | X_U, x_W), \qquad (3)$$

whenever $p(X_b, X_U, x_W) > 0$.

Example 2. *Figure 2(a) shows the graph of Example 1, induced from a log-linear model. Notice that the features of Example 1 encode the CSI $I(X_a, X_b \mid X_f = 1)$, but it is obscured in the graph. Alternatively, such CSI can be graphically represented if we use two graphs, one for each value of X_f. For this, Figure 2(b) shows the graph induced from the features with $X_f = 1$ which encodes $I(X_a, X_b \mid X_f = 1)$, and Figure 2(c) shows the graph induced from the features with $X_f = 0$ which encodes $\neg I(X_a, X_b \mid X_f = 0)$. In these figures, gray nodes correspond to an assignment of a variable.*

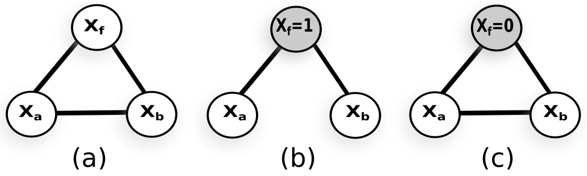

Figure 2: (a) The graph induced from the features in Example 1. (b) graph induced from the features with $X_f = 1$ in Example 1. (c) graph induced from the features with $X_f = 0$ in Example 1. Gray nodes correspond to an assignment of a variable.

Notice that the graph in Figure 2(a) cannot encode the CSI $I(X_a, X_b \mid X_f = 1)$, because it occurs only for a specific context and is absent in all the others. This is because the edges connect pairs of variables that are conditionally dependent even for a single choice of values of the other variables. Since a CSI is defined for a specific context, a set of CSIs cannot be encoded all together in a single undirected graph [19]. Nonetheless, both structures are encoded in the set of features of the example.

C. Independence-based approach for structure learning

The task of IB algorithms is learning a graph that encodes the independences from i.i.d. samples $\mathcal{D} = \{x^1, \dots, x^D\}$ of an unknown underlying distribution $p(X)$ [1]. For that, these algorithms perform a succession of statistical independence tests over $\mathcal{D}$ to determine the truth value of a conditional independence (e.g. Mutual Information [20], Pearson's χ^2 and $\mathcal{G}^2$ [21]), discarding all graphs that are inconsistent with the test. The decision of what test to perform is based on the independences learned so far, and varying with each specific algorithm.

A key advantage of these algorithms is that they guarantees to learn the correct underlying structure under three assumptions: (*i*) the underlying distribution is *graph-isomorph*, that is, the independences in $p(X)$ can be encoded by a graph; (*ii*) the underlying distribution is positive, that is $p(x) > 0$ for all $X = x$; and (*iii*) the outcomes of statistical independence tests are correct, that is the independences learned are a subset of the independences present in $p(X)$. Another advantage of using IB algorithms is its computational efficiency, due to its polynomial running time [1], and also due to the avoiding of the need of performing parameters learning. The efficiency is gained because the computational cost of a test is proportional to the number of rows in $\mathcal{D}$, and the number of variables involved in the tests.

Perhaps the best known algorithm that follows this approach is PC [16], which was created for learning the structure of Bayesian networks. PC is correct under the assumptions described above, but when the tests are not correct produce errors in removing edges, because the algorithm only tests for independence among two variables conditioning in subsets of the adjacencies of one of these variables. For learning Markov networks, the first algorithm that follows the IB approach is GSMN [22], an efficient algorithm that computes only $O(n^2)$ tests, constructing the structure by learning the adjacencies of each variable; using the *Grow-Shrink* algorithm [23]. A more recent algorithm that improves over GSMN is IBMAP-HC [24], which learns the structure by performing a hill-climbing search over the space of graphs looking for the one which maximizes the IB-score, a score of the posterior probabilities of graphs $p(G \mid \mathcal{D})$. The hill-climbing search starts from the empty structure, adding edges until reaching a local maxima of $p(G \mid \mathcal{D})$. IBMAP-HC relaxes the assumption about the correctness of the statistical tests, improving over GSMN in sample complexity by reducing the cascade effect of incorrect tests.

III. Context-Specific Parent and Children algorithm

This section presents the CSPC (Context-specific Parent and Children) algorithm for learning Markov networks structures that encodes the CSIs present in data. CSPC encodes the CSIs by generalizing iteratively a set of features. For this, CSPC decomposes the search space of CSIs in two nested spaces: the space of the possible contexts, and for each context the space of all its possible CSIs. First, CSPC generates an initial set of features, and then searches over both spaces with two nested loops: an outer loop that explores the space of the contexts; and for each one, an inner loop that elicits from data a set of CSIs by using statistical tests, and generalizes the features according to the elicited CSIs.

The three key elements of CSPC are: *A)* the generation of the initial features, *B)* the elicitation of CSIs from data, and *C)* the features generalization for encoding the elicited CSIs.

A. The generation of the initial features

An initial set of features $\mathcal{F}$ must be generated as a starting point of the whole algorithm. One alternative is to generate the features that correspond with the initial fully connected graph in a similar fashion than PC, that is adding a feature for each possible complete assignment x of the variable X. In this case, the size of such initial set is exponential with respect to the number of variables. For this reason, CSPC uses a more optimal initial set of features, adding one feature for each unique example in $\mathcal{D}$. This is an often used alternative [6], [9], because it guarantees that the generalized features at the end of the algorithm match at least one training example.

B. Eliciting context-specific independences

For discovering all the CSIs present in the data, CSPC explores the set of complete contexts x found in the dataset, that is, one for each unique training example. Given a context $x \in \mathcal{D}$ and a set of features $\mathcal{F}$, CSPC decides what CSIs to elicit in a similar fashion than PC. For each variable X_a a Markov network is induced from the subset of features that satisfies with the context $x_{X \setminus X_a}$. Then, for each X_b adjacent to X_a in the induced graph a subset X_W of the adjacencies is taken in order to define the conditional independence $I(X_a, X_b \mid X_W)$. From such independence, a CSI is obtained by contextualizing the conditioning set X_W using the context x, that is $I(X_a, X_b \mid X_W = x_w)$. Finally, if the CSI is present in data then it is encoded in the current set of features.

For eliciting the CSIs in data, we propose a straightforward adaptation of a traditional (non-contextualized) independence test. A similar adaptation is proposed in [8]. The central idea of the adaptation is that an arbitrary CSI $I(X_a, X_b \mid X_U, x_W)$ can be seen as a conditional independence $I(X_a, X_b \mid X_U)$ in the conditional distribution $p(X \setminus \{X_W\} \mid x_W)$. In this way, the CSI can be tested by using a non-contextualized test over a sample drawn from the conditional distribution $p(X \setminus \{X_W\} \mid x_W)$. In practice, such sample can be obtained from $\mathcal{D}$ as $\{x^j \in \mathcal{D} \colon x^j_W = x_W\}$, namely, the subset of datapoints where $X_W = x_W$.

C. Features generalization

The generalization of features is used by CSPC to encode the CSIs that are present in data. A specific CSI $I(X_a, X_b \mid x_W)$ can be encoded in the features $\mathcal{F}$ of a log-linear of $p(X)$ by factorizing those features that correspond with the conditional distribution $p(X \setminus \{X_W\} \mid x_W)$. Such factorization is done by using a recently proposed adaptation of the well known Hammersley-Clifford theorem for CSIs called the Context-Specific Hammersley-Clifford theorem [25]. The features that correspond with $p(X \setminus \{X_W\} \mid x_W)$ are the subset of features in $\mathcal{F}$ that satisfy the context x_W, denoted by $\mathcal{F}[x_W] \subseteq \mathcal{F}$. In this way, given the CSI $I(X_a, X_b \mid x_W)$ the features $\mathcal{F}[x_W]$ are factorized into two new sets of features: $\mathcal{F}'[x_W]$, obtained from $\mathcal{F}[x_W]$ but removing the variable X_a; and $\mathcal{F}''[x_W]$, obtained from $\mathcal{F}[x_W]$ but removing the variable X_b. Formally, for all $f'_j \in \mathcal{F}'[x_W]$, $\{X_a\} \notin V(f'_j)$; and for all $f''_j \in \mathcal{F}''[x_W]$, $\{X_b\} \notin V(f''_j)$.

Example 3. *Figure 3a shows an initial set of features $\mathcal{F}$ to be generalized in order to encode the CSI $I(X_a, X_b \mid X_f = 1)$. The generalization consists in factorizing the features $\mathcal{F}[X_f = 1]$, that is the set of features that are satisfied with the context $X_f = 1$ (Figure 3b). The factorization of these features results in two new sets of features: $\mathcal{F}'[X_f = 1]$ and $\mathcal{F}''[X_f = 1]$, shown in Figure 3c. The features in $\mathcal{F}'[X_f = 1]$ are obtained from $\mathcal{F}[X_f = 1]$ but removing X_b, and the features in $\mathcal{F}''[X_f = 1]$ are obtained from $\mathcal{F}[X_f = 1]$ but removing X_a Finally, the set of features which correctly encodes the CSI $I(X_a, X_b \mid X_f = 1)$ are shown in Figure 3d. Notice that the features in Figure 3d are the same set of features shown in Example 1.*

D. Overview

This section presents an explanation of CSPC that puts all the pieces together. The pseudocode is shown in Algorithm 1. As input, the algorithm receives the set of domain variables X, and a dataset $\mathcal{D}$. The algorithm starts by generating the initial set of features. Then, the space of the contexts is explored. For each context, the current set of features is generalized by using a generalization of the PC algorithm as a subroutine. This subroutine, described in Algorithm 2, consists in the elicitation of CSIs and the features generalization steps. As input, this subroutine receives the current set of features $\mathcal{F}$, the context x, the set of domain variables X, and the dataset $\mathcal{D}$. At the end, the features of a log-linear model are returned.

In Algorithm 2, the step of elicitation of CSIs follows the same strategy than PC, trying to find the independences on the smallest number of variables in the conditioning set. For this, the conditioning set for each variable X_a consists on subset of size k of the adjacencies $\operatorname{adj}(a)$, terminating when for all subsets W, $|W|$ is smaller than k. This is a strategy for avoiding the effect of incorrect tests, because in the practice the quality of statistical tests decreases exponentially with the number of variables that are involved [21].

$$\begin{array}{l}
f_1\ (X_a = 0\ X_b = 0\ X_f = 0)\\
f_2\ (X_a = 1\ X_b = 0\ X_f = 0)\\
f_3\ (X_a = 0\ X_b = 1\ X_f = 0)\\
f_4\ (X_a = 1\ X_b = 1\ X_f = 0)\\
f_5\ (X_a = 0\ X_b = 0\ X_f = 1)\\
f_6\ (X_a = 1\ X_b = 0\ X_f = 1)\\
f_7\ (X_a = 0\ X_b = 1\ X_f = 1)\\
f_8\ (X_a = 1\ X_b = 1\ X_f = 1)
\end{array}$$

(a) Initial set of features

$$\begin{array}{l}
\mathcal{F}[X_f = 1] = \{\\
\quad f_5(X_a = 0 X_b = 0 X_f = 1),\\
\quad f_6(X_a = 1 X_b = 0 X_f = 1),\\
\quad f_7(X_a = 0 X_b = 1 X_f = 1),\\
\quad f_8(X_a = 1 X_b = 1 X_f = 1)\}
\end{array}$$

(b) The features that satisfy with the context $X_f = 1$

$$\begin{array}{l}
\mathcal{F}'[X_f = 1] = \{\\
\quad f'_5(X_b = 0 X_f = 1),\\
\quad f'_6(X_b = 0 X_f = 1),\\
\quad f'_7(X_b = 1 X_f = 1),\\
\quad f'_8(X_b = 1 X_f = 1)\}
\end{array}$$

$$\begin{array}{l}
\mathcal{F}''[X_f = 1] = \{\\
\quad f''_5(X_a = 0 X_f = 1),\\
\quad f''_6(X_a = 1 X_f = 1),\\
\quad f''_7(X_a = 0 X_f = 1),\\
\quad f''_8(X_a = 1 X_f = 1)\}
\end{array}$$

(c) Factorization of the features $\mathcal{F}[X_f = 1]$

$$\begin{array}{l}
f_1\ (X_a = 0\ \ X_b = 0\ \ X_f = 0)\\
f_2\ (X_a = 1\ \ X_b = 0\ \ X_f = 0)\\
f_3\ (X_a = 0\ \ X_b = 1\ \ X_f = 0)\\
f_4\ (X_a = 1\ \ X_b = 1\ \ X_f = 0)\\
f'_5\ (X_a = 0\ X_f = 1)\\
f'_7\ (X_a = 1\ X_f = 1)\\
f''_5\ (X_b = 0\ X_f = 1)\\
f''_7\ (X_b = 1\ X_f = 1)
\end{array}$$

(d) Features generalized encoding $I(X_a, X_b \mid x_W)$

Figure 3: Example of feature factorization according to CSIs.

```
Algorithm 1: Context space exploration
Input: domain variables $X$, dataset $\mathcal{D}$
Output: features $\mathcal{F}$ generalized according to the CSIs learned
$\mathcal{F} \leftarrow$ Generate one feature for each unique example in $\mathcal{D}$
    $x \in \mathrm{Val}(X)$
foreach context $x \in \mathrm{Val}(X)$ do
    $\mathcal{F} \leftarrow$ PC ($\mathcal{F}$, $x$, $X$, $\mathcal{D}$)
Add atomic feature for each variable to $\mathcal{F}$
return $\mathcal{F}$
```

In Algorithm 3, the step of features generalization is made once the CSI has been elicited. In such step, for the input CSI $I(X_a, X_b \mid x_W)$ the current set of features $\mathcal{F}$ is partitioned in two sets: the set $\mathcal{F}'$ that are the features that satisfy with x_W, and the set $\mathcal{F}''$ that are the features that does not satisfy with x_W. Then, the satisfied features are factorized according to the Context-Specific Hammersley-Clifford theorem. Finally, a new set of features $\mathcal{F}$ is defined by joining $\mathcal{F}'$ and $\mathcal{F}''$.

IV. Experimental evaluation

To allow a proper experimental design with a range of well-understood conditions, we evaluated our approach on artificially generated datasets. For testing the effectiveness of our approach, we propose a specific class of deterministic

Algorithm 2: PC extended for features

```
Input: features $\mathcal{F}$, context $x$, domain variables $X$, dataset $\mathcal{D}$
Output: generalized features $\mathcal{F}$
$k \leftarrow 0$
repeat
    foreach $X_a \in X$ do
        $\mathrm{adj}(a) \leftarrow$ compute adjacencies from features that satisfy $x_{X \setminus X_a}$
        foreach $X_b \in \mathrm{adj}(a)$ do
            foreach $W$ subset of adj(a) $\setminus \{X_b\}$ s.t. $|W| = k$ do
                if $I(X_a, X_b \mid x_W)$ is true then
                    $\mathcal{F} \leftarrow$ Generalize $\mathcal{F}$ for the CSI $I(X_a, X_b \mid x_W)$
    $k \leftarrow k + 1$
until $|\mathrm{adj}(a) \setminus \{X_b\}| < k$;
return $\mathcal{F}$
```

Algorithm 3: Feature generalization

```
Input: features $\mathcal{F}$, a CSI $I(X_a, X_b \mid x_W)$
Output: generalized features $\mathcal{F}$
$\mathcal{F}' \leftarrow \mathcal{F}[x_W]$
$\mathcal{F}'' \leftarrow \mathcal{F} \setminus \mathcal{F}'$
$\mathcal{F}' \leftarrow$ factorize $\mathcal{F}'$ according to the Context-Specific Hammersley-Clifford
return $\mathcal{F}' \cup \mathcal{F}''$
```

models which presents a controlled number of CSIs. The evaluation consists in two parts. In the first part we show the potential of improvements that can be obtained in our experiment. In the second part we compare CSPC to two state-of-the-art IB algorithms: GSMN [22] and IBMAP-HC [24]. Additionally we also compare with PC [16] in order to highlight the improvements resulting from contextualizing it. Since PC was originally designed for learning Bayesian networks (directed graphs), we use PC omitting the step of edges orientation [26].

A. Datasets

We generated artificial data through Gibbs sampling on a class of models similar to Example 2, generalized to distributions with n discrete binary variables. We chose such models since they are a representative case of a distribution with a controlled number of CSIs. The aim is to demonstrate that learning such CSIs represents an important improvement in the quality of learned distributions, when compared with the alternative representation in graphs. In this scenario, the IB algorithms lead to excessively dense graphs (the fully connected ones), which obscures the underlying CSIs. We considered models with $n \in \{6, 7, 8\}$ variables and maximum cliques of the same size. Since the complexity of structure learning grows exponentially with the size of its maximum clique (a.k.a. treewidth), in the literature the algorithms are typically tested on models with maximum cliques of size at most 6 [8], [18].

For each n, the underlying structure is a fully connected graph with $(n-1)$ nodes, plus a flag node X_f. In this model, all pairs between the variables $X \setminus \{X_f\}$ are context-specific independent, given the context $X_f = 1$. Instead, when $X_f = 0$ the variables remain dependent. In this way, for n variables the underlying structure contains $\frac{(n-1)\times(n-2)}{2}$ contextual independences in the form $I(X_a, X_b \mid X_f = 1)$, for all $X_a, X_b \neq X_f \in X$. Given such structure, we defined a log-linear model that contains two sets of features: *i)* a set of *pairwise* features which encodes the dependence between X_f and the rest of variables $X \setminus \{X_f\}$, and *ii)* a set of *triplet* features over the variables X_a, X_b, X_f. For the resulting features we generated 10 different models, varying in its numerical parameters. Such parameters were generated to satisfy the log-odds ratio, in order to set strong dependencies in the model [3]–[5], [21]. In this way, the parameters of the pairwise features X_a, X_b were forced to satisfy the following ratio: $\varepsilon = \log\left(\frac{w_0\phi(X_a=0,X_f=0)w_1\phi(X_a=1,X_f=1)}{w_2\phi(X_a=0,X_f=1)w_3\phi(X_a=1,X_f=0)}\right), \forall X_a \in X \setminus \{X_f\}$, where w_0, w_2 are symmetric to w_1, w_3, respectively ($w_0 = w_1$ and $w_2 = w_3$). Since this ratio has 2 unknowns we choose w_2 sampled from $\mathcal{N}(0.5; 0.001)$, and w_0 is solved. The parameters for the triplet features were forced to satisfy the CSI $I(X_a, X_b \mid X_f = 1)$. When $X_f = 0$ the parameters were generated using the same procedure used for the pairwise features. When $X_f = 1$ the parameters were forced to satisfy the following factorization: $\phi(X_a, X_b, X_f = 1) = w_0\phi(X_a = 0, X_f = 1) \cdot w_1\phi(X_a = 1, X_f = 1)$, where w_0 and w_1 are the same than the pairwise features already defined. In our experiments we set $\varepsilon = 1.0$. The datasets were generated by sampling from the log-linear models using Rao-Blackwellized Gibbs sampler [1] with 10 chains, 100 burn-in and 1000 sampling.

B. Methodology

We used the synthetic datasets explained above to learn the structure and parameters for all the algorithms. Our synthetic data, together with an executable version of CSPC and the competitors is publicly available[2]. For a fair comparison, we use Pearson's χ^2 as the statistical independent test for all the algorithms, with a significance level of $\alpha = 0.05$. The IBMAP-HC algorithm alternatively only works by using the Bayesian test of Margaritis [23]. For each particular dataset we evaluated the algorithms on training set sizes varying from 500 to 40000, in order to obtain a number of samples sufficient for satisfying the CSIs of the underlying distribution proposed.

We report the quality of learned models using the Kullback-Leibler divergence (*KL*) [27]. The KL is defined as $KL(p \mid\mid q) = \sum_x p(x) ln \frac{p(x)}{q(x)}$, measuring the information

[1] Gibbs sampler is available in the open-source Libra toolkit http://libra.cs.uoregon.edu/

[2] http://dharma.frm.utn.edu.ar/papers/cspc

lost when the learned distribution $q(X)$ is used to approximate the underlying distribution $p(X)$. KL is equal to zero when $p(X) = q(X)$. The better the learned models, the lower the values of the KL measure. Since these algorithms only learn the structure of a Markov network, the complete distribution is obtained by learning its numerical parameters. For the case of IB algorithms, the features are induced from the maximum cliques of the graph learned. For learning its parameters we computed the pseudo-loglikelihood using the available version in the Libra toolkit. We use pseudo-loglikelihood without regularization to avoid sparsity in the final model, because we are interested in measuring the quality of the structure learning step.

C. Results

Our first experiment shows the potential improvement that can be obtained in our generated datasets in terms of KL over the generated data. For this we measure in Figure 4 the KLs obtained by learning the parameters for three proposed structures: *i)* the empty structure, *ii)* the fully connected structure, and *iii)* the underlying structure. The distribution learned from the empty structure informs us about the impact of encoding incorrect independences in the KL measure. Consequently, the fully connected structure shows the impact in the KL measure that can be obtained with incorrect dependences that are obscuring the real CSIs present in data. The underlying structure contains the features which exactly encodes the CSIs of the proposed model, as described in Section IV-A. The figure shows the average and standard deviation over our 10 generated datasets for training set sizes varying from 500 to 40000 (X-axis), for different domain sizes $6, 7$ and 8. In order to better show differences among the KLs we show it in log scale.

In these results, we see empirically that the KL of the distribution obtained by learning the parameters for the underlying structure is always significantly better than the KL obtained by using the empty and fully structures. Notice that the KL is an (expected) logarithmic difference, representing differences in orders of magnitude in the non-logarithmic space. For our results, these differences are up to 2 orders of magnitude in the cases of $n \in \{6, 7\}$, and 5 orders of magnitude in the case of $n = 8$. However, there is not a trend of the KL to be zero by varying the size of training data. This is because Gibbs sampler is not an exact method for the generation of the training data. In this result also can be seen that as well as n increases, the KL of the fully structure is better than the KL of the empty structure.

In our second experiment we compare the KLs obtained by CSPC, GSMN, IBMAP-HC, and PC. These results are shown in Figure 5, for the same datasets of the experiment shown in Figure 4. CSPC is the more accurate algorithm in all the cases, with lower KL, near to 1 (the KL value of the underlying structure in Figure 4). The differences in KL between CSPC against its competitors is up to 2 orders of magnitude for $n = 6$ and $n = 7$, and up to 5 orders of magnitude against IBMAP-HC for $n = 8$.

Since the KL is clearly affected by the quality of the structure, we wanted to determine whether or not their actual structures are correct. We did this by reporting the average feature length of the learned models, since it is a known value for our underlying model in this experiment. This is a statistical measure useful for analyzing the structural quality of log-linear models, as shown in several recent works [7], [9], [28]. Figure 6 reports these values for the same experiment shown above. The horizontal line in the graphs shows the exact feature length of the underlying structure (2.80 for $n = 6$, 2.83 for $n = 7$, and 2.85 for $n = 8$). CSPC perform better in all the cases, showing always the nearest number of average feature length to the horizontal line. This is consistent with the KL results shown in Figure 5. GSMN and PC increases in the average number of features length as well as the number of datapoints grows, for all the domain sizes. This trend is due because they are learning more dense structures as well as the number of datapoints grows, reaching the fully structure. For example, in the case of $n = 7$, the GSMN and PC algorithms shows a trend to reach the fully structure, and the KLs shown in Figure 5 are similar to the KL of the fully structure shown in Figure 4. Also, as well as n increases, the difference on the average feature length between CSPC and its competitors also increases. This is also consistent with the results shown in Figure 4. A surprising result is shown for IBMAP-HC, which does not show the same trend than GSMN and PC. It can be seen that for lower number of datapoints ($\mathcal{D} < 5000$) the algorithm learns fully structures (the average feature length is equal to n in the three cases). However, for higher number of datapoints ($\mathcal{D} \geq 5000$) the algorithm learns the empty structure, with average feature length equal to 1 (the empty structure contains the atomic features). We argue this is due to the Bayesian nature of IBMAP-HC, which works by optimizing the posterior probability of structures $p(G \mid \mathcal{D})$ with a hill-climbing search. When using a large amount of data, IBMAP-HC seems very prone to getting stuck in the empty structure as a local minima with $p(G \mid \mathcal{D}) = 0$ for almost all the structures, except the correct one with $p(G \mid \mathcal{D}) = 1$.

Finally, as an additional result, we show in Table I the average feature length of the features that are satisfied with the value of the flag variable X_f. This result is shown for all the algorithms, running for an increasing number of variables from $n = 4$ to 8 with a fixed number of 3000 datapoints, and discriminating in different columns the value of X_f. As expected, the values for CSPC for $X_f = 0$ are more near to 3, and more near to 2 for $X_f = 1$, in comparison with the rest of the competitors. In summary, the above results support our theoretical claims and demonstrate the efficiency of CSPC for learning distributions with CSIs.

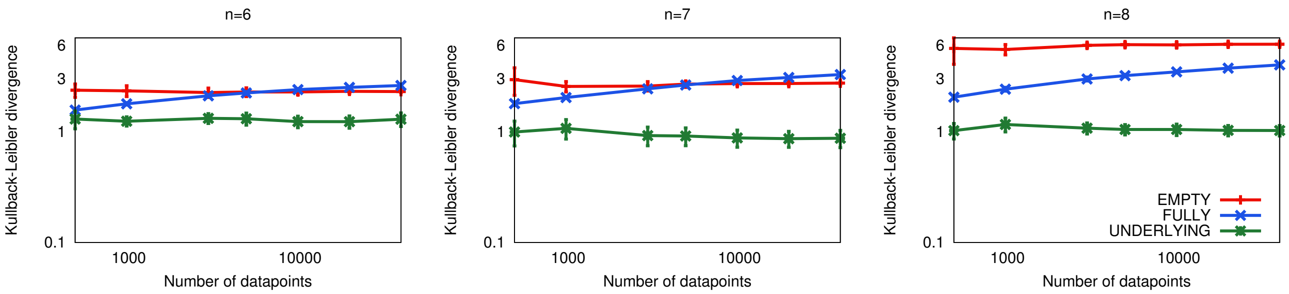

Figure 4: Potential improvements in KL obtained by learning parameters for the underlying structure, the fully and the empty structures. Average and standard deviation over ten repetitions for increasing number of datapoints in the training set for domain sizes 6 (left), 7 (center) and 8 (right).

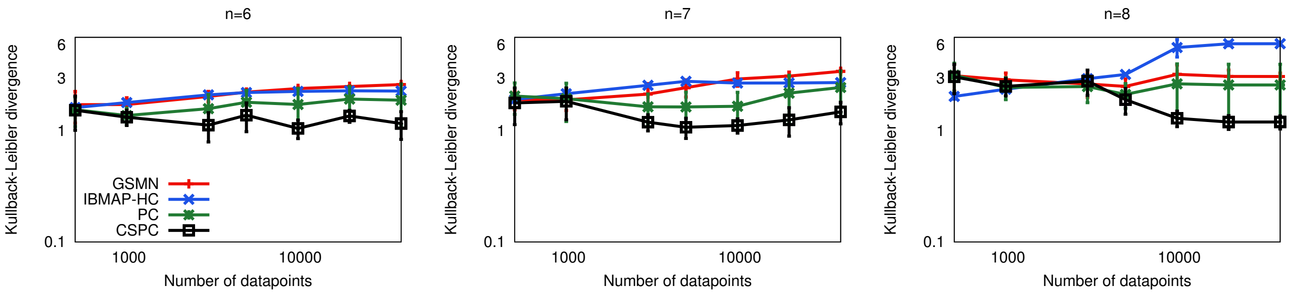

Figure 5: Comparison of KLs obtained by learning parameters for CSPC, GSMN, IBMAP-HC and PC. Average and standard deviation over ten repetitions for increasing number of datapoints in the training set for domain sizes 6 (left), 7 (center) and 8 (right).

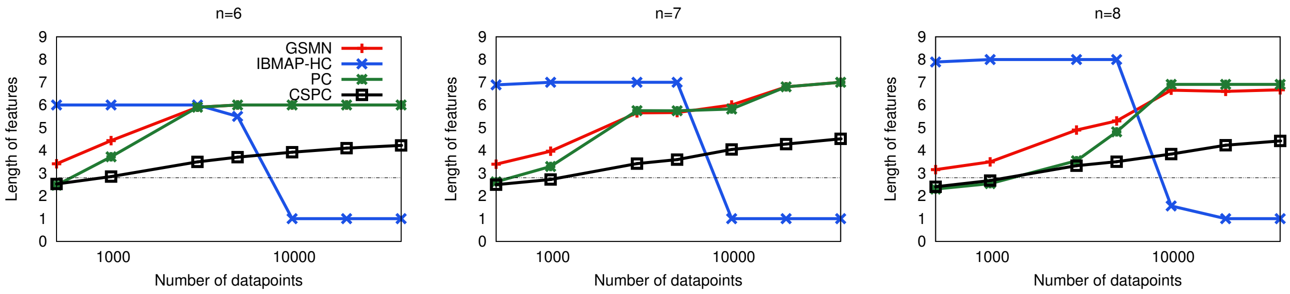

Figure 6: Comparison of the average feature length obtained for CSPC, GSMN, IBMAP-HC and PC. Average and standard deviation over ten repetitions for increasing number of datapoints in the training set for domain sizes 6 (left), 7 (center) and 8 (right). The average feature length of the solution underlying structure is the horizontal line.

	$X_f = 0$				$X_f = 1$			
n	GSMN	IBMAP-HC	PC	CPSC	GSMN	IBMAP-HC	PC	CPSC
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	1.85
5.00	5.00	4.60	5.00	4.79	5.00	4.60	5.00	1.88
6.00	6.00	5.50	6.00	4.37	6.00	5.50	6.00	1.93
7.00	5.00	7.00	1.10	4.01	5.00	7.00	1.10	1.95
8.00	3.00	8.00	1.00	3.54	3.70	8.00	1.00	1.87

Table I: Number of features learned for increasing n, and using $\mathcal{D} = 3000$.

V. Conclusions

This paper proposed CSPC, an independence-based algorithm for learning a set of features, instead of a graph. CSPC overcomes some of the inefficiency of traditional IB algorithms by learning CSIs from data and representing them in a log-linear model. CSPC proceeds by generalizing iteratively a set of initial features in order to represent the CSIs present in data, exploring the possible contexts, eliciting from data a set of CSIs usings statistical tests, and generalizing the features according to the elicited CSIs. Experiments in a synthetic case show that this approach is more accurate than the state-of-the-art IB algorithms, when the underlying distribution contains CSIs. Directions of future work include: adapting more efficient IB algorithms for learning CSIs; validation in real world datasets; comparison against state-of-the-art non-independence-based approaches [6]–[9]; adding Moore and Lee's AD-trees [29] for speeding up the execution of statistical tests, etc.

References

[1] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, ser. Adaptive Computation and Machine Learning Series. MIT Press, 2000.

[2] F. Bromberg, D. Margaritis, and H. V., "Efficient Markov Network Structure Discovery Using Independence Tests," *JAIR*, vol. 35, pp. 449–485, July 2009.

[3] P. Gandhi, F. Bromberg, and D. Margaritis, "Learning Markov Network Structure using Few Independence Tests," in *SIAM International Conference on Data Mining*, 2008, pp. 680–691.

[4] D. Margaritis and F. Bromberg, "Efficient Markov Network Discovery Using Particle Filter," *Comp. Intel.*, vol. 25, no. 4, pp. 367–394, 2009.

[5] F. Bromberg, F. Schlüter, and A. Edera, "Independence-based MAP for Markov networks structure discovery," in *International Conference on Tools with Artificial Intelligence*, 2011, http://ai.frm.utn.edu.ar/fschluter/p/11d.pdf.

[6] J. Davis and P. Domingos, "Bottom-up learning of markov network structure," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 271–280.

[7] D. Lowd and J. Davis, "Learning markov network structure with decision trees," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 334–343.

[8] V. Gogate, W. Webb, and P. Domingos, "Learning efficient markov networks," in *Advances in Neural Information Processing Systems*, 2010, pp. 748–756.

[9] J. Van Haaren and J. Davis, "Markov network structure learning: A randomized feature generation approach," in *Proceedings of the Twenty-Sixth National Conference on Artificial Intelligence. AAAI Press*, 2012.

[10] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller, "Context-specific independence in Bayesian networks," in *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1996, pp. 115–123.

[11] D. Poole and N. L. Zhang, "Exploiting contextual independence in probabilistic inference," *J. Artif. Intell. Res.(JAIR)*, vol. 18, pp. 263–313, 2003.

[12] A. Fridman, "Mixed markov models," *Proceedings of the National Academy of Sciences*, vol. 100, no. 14, pp. 8092–8096, 2003.

[13] C. Benedek and T. Szirányi, "A mixed markov model for change detection in aerial photos with large time differences," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[14] D. Fierens, "Context-specific independence in directed relational probabilistic models and its influence on the efficiency of Gibbs sampling," in *European Conference on Artificial Intelligence*, 2010, pp. 243–248.

[15] Y. Wexler and C. Meek, "Inference for multiplicative models," in *Uncertainty in Artificial Intelligence*, 2008, pp. 595–602.

[16] P. Spirtes and C. Glymour, "An algorithm for fast recovery of sparse causal graphs," *Social Science Computer Review*, vol. 9, no. 1, pp. 62–72, 1991.

[17] J. M. Hammersley and P. Clifford, "Markov fields on finite graphs and lattices," 1971.

[18] S. Lee, V. Ganapathi, and D. Koller, "Efficient structure learning of Markov networks using L1-regularization," in *Neural Information Processing Systems*. Citeseer, 2006.

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

[20] T. M. Cover and J. A. Thomas, *Elements of information theory*. New York, NY, USA: Wiley-Interscience, 1991.

[21] A. Agresti, *Categorical Data Analysis*, 2nd ed. Wiley, 2002.

[22] F. Bromberg, D. Margaritis, V. Honavar *et al.*, "Efficient markov network structure discovery using independence tests," *Journal of Artificial Intelligence Research*, vol. 35, no. 2, p. 449, 2009.

[23] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," DTIC Document, Tech. Rep., 2000.

[24] F. Bromberg, F. Schluter, and A. Edera, "Independence-based map for markov networks structure discovery," in *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*. IEEE, 2011, pp. 497–504.

[25] A. Edera, F. Bromberg, and F. Schlüter, "Markov random fields factorization with context-specific independences," *arXiv preprint arXiv:1306.2295*, 2013.

[26] M. Kalisch and P. Bühlmann, "Estimating high-dimensional directed acyclic graphs with the pc-algorithm," *The Journal of Machine Learning Research*, vol. 8, pp. 613–636, 2007.

[27] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[28] D. Lowd and A. Rooshenas, "Learning markov networks with arithmetic circuits," *The Journal of Machine Learning Research*, vol. 31, pp. 406–414, 2013.

[29] A. Moore and M. S. Lee, "Cached suficient statistics for e cient machine learning with large datasets," *Journal of Artificial Intelligence Research*, vol. 8, pp. 67–91, 1998.