Reasoning Cyber Intrusions: A Copula Bayesian Netwroks Approach

Abstract

...

1. Introduction

The identification of causal relationships remains an important topic in the analysis of dynamic processes. Bayesian Networks (Heckerman, 1995) (BNs) are powerful tools to uncover such relationships by offering a graphical structure along with conditional probability distributions/tables that reflect the interdependencies of included variables. Standard Bayesian network learning methods always include two main challenges: parameter learning and structure learning. Although a lot of effort has been put into this area, many problems remain, such as the computational effort required by many approaches, inaccuracies in structure inferences and non-trivial parameter estimation problems. Copula Bayesian networks (CBNs), recently proposed by Elidan (2010a), incorporate ideas from Copula theory to offer a simplified parameter estimation strategy. In this paper, we develop a structure learning approach for CBNs. In a first step, we use Copula functions to reduce the prior parameters to be estimated by the segregation of univariate marginals from the multivariate distribution. Subsequently, we base the structure inference on the partial inverse correlation matrix (PICM). In experiments, we show that this strategy is faster than competing approaches and also results in Copula Bayesian networks (CBNs) of good accuracy.

This paper is organized as follows. First, an overview of related work is presented. Next, Section 3 gives a short introduction to Copulas and then describes the PICM-CBN algorithm in detail, which is subsequently evaluated in Section 4. The paper closes with a discussion.

2. Related Work

Copula Bayesian network (CBN) is a powerful tool of analysing multivariate model. The main advantage of CBNs is the high flexibility of representing multivariate distributions by choosing various univariate marginals, meanwhile leveraging the graphical representation of BNs. Prior work (Elidan, 2010a) has studied the basic problem of parameter learning for CBNs. It has shown that CBNs are also elegant models dealing with a large number of missing observations (Elidan, 2010b), where a lower bound for a log-likelihood function is proposed for efficient inference. However, the structure learning problem in CBNs has not ever been very well studied, which is considered as the most challenging problem of learning a Bayesian network. In terms of structure learning in BNs, two most commonly used algorithms are: First, the PC algorithm (Spirtes et al., 2001) which belongs to the category of constraint based methods (Spirtes et al., 2000). And second, scoring function based heuristic methods, especially based on the Bayesian information criterion (BIC) (Heckerman, 1995). The PC algorithm starts from a completely connected graph, edges are removed if the corresponding independencies are given, which usually requires a large amount of conditional independence tests. The BIC score based heuristic method generates the network structure in a greedy strategy, so the global minimum is not guaranteed. Other structure learning methods, such as the Sparce Candidate (SC) algorithm (Friedman, 1999), best-first search (Korf, 1993), all suffer from either structural inaccuracy or heavy computational effort. In this paper, we suggest the use of the partial inverse correlation matrix, which largely reduces the amount of conditional independence tests so that the learning is extremely fast. Moreover, the estimated parameter of the Copula function (Gaussian Copula) can be used further as the input for the structure learning. Furthermore, even with a large amount of missing values in training data, the estimated parameters of the Copula function still result in a precise structure inference.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

3. Methodology

This section introduces Copula Bayesian Networks first, and then the structure inference leveraging the partial inverse correlation matrix. At last, these two components are integrated to form the PICM-CBN algorithm.

3.1. Copula Bayesian Network

The Copula function is defined as a multivariate probability distribution within the domain of a N-dimensional unit hypercube. It can be selected as the prior for parameter estimation in Bayesian networks. A framework for Copula Bayesian networks (Elidan, 2010a) was proposed by combining Copula theory and BNs.

3.1.1. Copulas

A Copula ((Sklar, 1959), (Nelsen, 2006)) is a function C linking univariate marginals to generate a multivariate distribution. In domain $\mathcal{X} = \{X_1, \dots, X_N\}$ consisting of an N real-valued random variables. Let $F_{\mathcal{X}}(x) \equiv P(X_1 \leq x_1, \dots, X_N \leq x_n)$ be a cumulative joint probability distribution over $\mathcal{X}$ where $x = \{x_1, \dots, x_n\}$ is an assignment of variables X.

Definition 3.1 (Copulas) *Let $U_1, \dots, U_N$ be real random variables marginally uniformly distributed over $[0,1]$. A Copula function C is a cumulative joint probability function: $[0,1]^N \to [0,1]$.*

$$C(u_1, \dots, u_N) = P(U_1 \leq u_1, \dots, U_N \leq u_N)$$

From Definition 3.1, a Copula function C can be viewed as a probability function of points distributed in a N-dimensional unit hypercube. Copulas are important because of Sklar's theorem (Sklar, 1959) as described below.

Theorem 3.2 (Sklar 1959) *Let $F(x_1, \dots, x_N)$ be any cumulative multivariate distribution over real-valued random variables, then there exists a copula function C such that*

$$F(x_1, \dots, x_N) = C(F(x_1), \dots, F(x_N)),$$

where $F(X_i)$ is marginal cumulative density distribution of variable X_i and if each $F(X_i)$ is continuous, then the Copula is unique.

Moreover, if $F(X_1, \dots, X_N)$ has N-order partial derivatives, the joint density function can be obtained by

$$\begin{aligned} f(x) &= \frac{\partial^N C(F(x_1), \dots, F(x_N))}{\partial F(x_1) \dots F(x_N)} \prod_i f(x_i) \\ &= c(F(x_1), \dots, F(x_N)) \prod_i f(x_i), \end{aligned} \tag{1}$$

where $c(F(x_1), \dots, F(x_N))$ is called copula density function. Using Equation 1, it is very easy to obtain a joint density distribution once the univariate marginals are estimated. This theorem gives the importance of Copulas that, for any multivariate distribution given its marginals, we can find a Copula distribution function to formulate its dependency structure taking univariate marginals as Copulas' arguments.
A simple example is the Gaussion Copula which is widely used because of its simplicity and practical applications. Suppose the correlation matrix Σ, a Gaussian Copula can be constructed simply by inverting Sklar's theorem (Sklar, 1959)

$$C(\{F(x_i)\}) = \Phi_\Sigma\left(\phi^{-1}(F(x_1)), \dots, \phi^{-1}(F(x_N))\right), \tag{2}$$

where ϕ is standard normal cumulative distribution, Φ_Σ is standard normal cumulative distribution with correlation matrix Σ. Using Sklar's Theorem 3.2 and Equation 1, the multivariate Gaussian density distribution can be easily obtained. The correlation matrix Σ is the only parameter to be estimated when univariate marginals are known from data observations.

3.1.2. Combining Copulas with Bayesian Networks

In Bayesian networks, the Markov property allows the network to be split into local terms where a variable is only conditioned on its parents so that the joint probability distribution can be expressed as a product of a collection of local conditional probability distributions. This factorization can also be applied to Copulas to form the conditional Copula density functions.

Remark 3.3 (Conditional Copula) *Let x denote a variable and $y = \{x_1, \dots, x_K\}$ the parents of x, $f(x \mid y)$ the conditional density function and $f(x)$ the marginal density of x. Then there exists a Copula density function $c(F(x), F(y_1), \dots, F(y_K))$ such that:*

$$f(x|y) = R_c(F(x), F(y_1), \dots, F(y_K)),$$

where R_c is the Copula ratio

$$\begin{aligned}R_c(F(x),&F(y_1),\ldots,F(y_K)) \\ &\equiv \frac{c(F(x),F(y_1),\ldots,F(y_K))}{\int c(F(x),F(y_1),\ldots,F(y_K))f(x)dx} \\ &= \frac{c(F(x),F(y_1),\ldots,F(y_K))}{\frac{\partial^K C(1,F(y_1),\ldots,F(y_K))}{\partial F(y_1)\ldots\partial F(y_K)}}\end{aligned} \tag{3}$$

and R_c is defined to be 1 when $\mathbf{y}=\emptyset$.

This implies a parametric form of a conditional probability distribution $f(x\,|\,\mathbf{y})$ given a Copula density function $c(F(x),F(y_1),\ldots,F(y_K))$ and a marginal density function $f(x)$.

Remark 3.4 (Decomposition of Copulas) *Given a directed acyclic graph $\mathcal{G}$ encoding conditional independencies over $\mathcal{X}$, and let $f(x) = c\left(F\left(x_1\right),\ldots,F\left(x_N\right)\right)\prod_i f\left(x_i\right)$ be the Copula density function which is strictly positive for every value of $\mathcal{X}$. If $f(x)$ is decomposable according to $\mathcal{G}$, then the Copula density $c\left(F\left(x_1\right),\ldots,F\left(x_N\right)\right)$ can also be decomposed according to $\mathcal{G}$*

$$c\left(F\left(x_1\right),\ldots,F\left(x_N\right)\right) = \prod_i R_{c_i}(F(x_i),\{F(\mathbf{Pa_{ik}})\}),$$

where c_i is a local Copula density function defined over each local term decomposed from $\mathcal{G}$ for each variable x_i, conditioned on its corresponding parents $\mathbf{Pa_i}$. Therefore, the Copula density function can be factorized in a similar way as a Bayesian network. Given the set of univariate marginals for all variables, an elegant framework for Copula Bayesian Networks was proposed.

Definition 3.5 (Copula Bayesian Network) *A Copula Bayesian Network (CBN) (Elidan, 2010a) is a triplet $\mathcal{C} = (\mathcal{G},\Theta_C,\Theta_f)$ encoding the joint density $f_{\mathcal{X}}(x)$. Θ_C is a set for all local Copula densities $c_i(F(x_i),\{F(\mathbf{Pa_{ik}})\})$ and Θ_f is the set of parameters representing the univariate marginals $f(x_i)$. Then $f_{\mathcal{X}}(x)$ can be parameterized as*

$$f_{\mathcal{X}}(x) = \prod_i R_{c_i}(F(x_i),\{F(\mathbf{Pa_{ik}})\})f(x_i)$$

Note that Copulas separate the choices of marginal parametric form from the joint probability distribution, thus more general and accurate non-parametric density estimations are allowed to be applied on univariate marginals. By sharing the global univariate marginals, the hypothesis space on parameters has been largely reduced, which enables efficient estimation of BN parameters.

3.2. The *PICM-CBN* Algorithm

A correlation matrix neither reflects the causality nor the Markov properties of Bayesian Networks, it only quantifies the pairwise correlation of variables. In order to infer more information among variables, a more specific measure of correlations is needed. The inverse correlation matrix (Whittaker, 2008) provides more correlative information. It reflects directly the independences in terms of partial covariance which can be used to construct an independence graph satisfying the Markov properties.

3.2.1. Partial Inverse Correlation Matrix

Lemma 3.6 *Given a p-dimensional vector X and a q-dimensional vector Y. Denote the inverse variance matrix $var(X,Y)^{-1}$ by D partitioned by*

$$D = \begin{pmatrix} D_{XX} & D_{XY} \\ D_{YX} & D_{YY} \end{pmatrix}$$

where $D_{YY} = var(Y\,|\,X)^{-1}$ is the reciprocal of a partial variance of Y given X.

Lemma 3.7 *Scale the inverse variance matrix D as it has unit off-diagonals. Each off-diagonal element in the inverse variance matrix D can be obtained by:*

$$D_{ij} = \frac{-D_{ji}}{\sqrt{D_{ii} * D_{jj}}}$$

Moreover, it is the negative of the partial correlation between the two corresponding variables, conditioned on all the rest.

This leads to a particularly important corollary about the zero partial covariance.

Corollary 3.8 *The off-diagonal block D_{XY} of an inverse variance D is zero if and only if $cov(Y\,|\,X) = 0$ and also $corr(Y\,|\,X) = 0$*

This relates directly to the scaled inverse correlation matrix such that each zero off-diagonal block indicates that variables in Y are independent given X. In practice, the zero-approximation of an off-diagonal element in the inverse correlation matrix implies the conditional independence. This constructs a moral graph.

3.2.2. Detriangulation of Moral Graph

Definition 3.9 (Moral Graph) *A moral graph is the undirected version of the original directed graph bridging any pair of nodes which converge to a common node.*

This converged node is also called *collider*, and the corresponding V-structure is depicted in Figure 1.

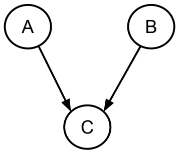


Figure 1. V-structure: Two non-adjacent nodes converging to a common node become correlated. In the corresponding moral graph, they are bridged by an additional edge.

Algorithm 1 DetriangulateMoralGraph

```
Input: $\mathcal{G} = (V, E)$, Correlation $\Sigma$, threshold $\sigma$
Output: A partially directed graph $\mathcal{G}^{\prec}$
for each $e = (p, q)$ in $E$ do
    $C \leftarrow \{v_i \in V \mid v_i$ and $e$ form a triangle$\}$
    for colliders $\leftarrow$ each $k$-combination in $C$ do
        /* $k = 1, 2, \dots, |C|$ */
        $N \leftarrow \{v_i \in V \mid (v_i, p) \in E$ or $(v_i, q) \in E\}$
        $\hat{N} \leftarrow \{\mathrm{N} \setminus \mathrm{colliders}\}$
        $R \leftarrow \Sigma_N^{-1}$
        $S \leftarrow \Sigma_{\hat{N}}^{-1}$
        /* $\Sigma_S$ indicates the partial correlation matrix
        from $\Sigma$ with respect to set $S$ */
        if $S_{pq} < \sigma$ then
            if $R_{pq} > \sigma$ then
                $E \leftarrow \{E \setminus e\}$
                orient $p$ and $q$ to all nodes in colliders
                break
            end if
        end if
    end for
end for
return $\mathcal{G}^{\prec} \leftarrow \mathcal{G}$
```

For each edge in a moral graph, it requires determining whether it is a bridged edge due to colliders or not. Suppose there is an edge e_{ij} between node i and j, denote the Markov blanket of both nodes as M_{ij} and its potential colliders as C_{ij} forming triangles with e_{ij}. Given M_{ij}, nodes i and j are d-separated from the other nodes. The partial inverse correlation matrix regarding the Markov blanket M_{ij} encodes the conditional independences of i and j. If now the colliders for nodes i and j are excluded from M_{ij}, then the inverse correlation matrix should be changed accordingly where additional information incurred by colliders will be now removed, in other words, e_{ij} is a bridged edge. This can be done simply by comparing inverse correlation matrix regarding $\{M_{ij} \setminus C_{ij}\}$ with the one containing C_{ij}. Discovering the colliders will also orient nodes i and j towards its collider set C_{ij}. The pseudo-algorithm for the detriangulation is shown in Algorithm 1. Since a bridged edge could be due to multiple colliders, the algorithm iterates through k-combinations of colliders (k starts from 1 to the maximal number of potential colliders) until the real colliders are found, unless it is not a bridged edge. Note that the algorithm could be computationally intensive when the graph is fully connected.

3.2.3. Constraint Propagation

All the V-structures discovered from detriangulation will impose underlying constraints on the entire network. These constraints can be propagated through the network conforming to the following rules (Pearl, 2000):

- R_1: Orient $a - b$ into $a \rightarrow b$ whenever there is an arrow $c \rightarrow a$ and b and c are not adjacent. (No new V-structure).
- R_2: Orient $a - b$ into $a \rightarrow b$ whenever there is a path $a \rightarrow c \rightarrow b$. (Preserve the acyclicity)
- R_3: Orient $a - b$ into $a \rightarrow b$ whenever there are two chains $a - c \rightarrow b$ and $a - d \rightarrow b$ such that c and d are not adjacent. (Three-fork structure)
- R_4: Orient $a - b$ into $a \rightarrow b$ whenever there are two chains $a - c \rightarrow d$ and $c \rightarrow d \rightarrow b$ and c and b are not adjacent.

Moreover, R_4 is not required, if the starting orientation is limited to V-structures. After constraint propagation, a maximally directed acyclic graph is formed. It does not require all the edges being directed. All the rules conform to either preserving the acyclicity or avoiding new V-structures. The pseudo-algorithm for constraint propagation is illustrated in Algorithm 2. It is very important to be aware of the order of applying these rules. The acyclicity should be the first and foremost of all the rules to be satisfied. Each time when an unknown edge is oriented, it should be followed by checking the acyclicity, or a cyclic graph could be created unintentionally.

Algorithm 2 ConstraintPropagation

```
Input: a partially directed graph $\mathcal{G}^{\prec}$
Output: a maximally oriented graph $\mathcal{G}^{\prec}$
while $\mathcal{G}^{\prec}$ is changed do
  if Edge$(X, Y)$ is undirected and $\exists$ directed path from $X$ to $Y$ then
    set $X \to Y$ /* preserve acyclicity */
    break
  end if
  if $\exists$ node $X$ where $Y \to X \leftrightarrow Z$ then
    set $Y \to X \to Z$ /* no new collider */
    break
  end if
  if $\exists$ an undirected edge $X \leftrightarrow Y$ and $\exists$ nonadjacent $Z$ and $W$ that $X \leftrightarrow Z \to Y$ and $X \leftrightarrow W \to Y$ then
    orient as $X \to Y$ /* three-fork structure */
    break
  end if
end while
```

Algorithm 3 PICM-CBN Learning

```
Input: Dataset $D$, threshold $\sigma$
Output: All equivalent DAGs
construct fully connected graph $\mathcal{G}$
$marginals \leftarrow$ estimate marginals for each variable
$\Sigma \leftarrow$ parameter estimation of Gaussian Copula
$\Sigma^{-1} \leftarrow$ inverse correlation matrix $\Sigma$
for each entry $e = \Sigma^{-1}_{ij}$ do
  if $e < \sigma$ then
    remove $edge\,(i, j)$ from $\mathcal{G}$
  end if
end for
set Moral graph $\hat{\mathcal{G}} \leftarrow \mathcal{G}$
$\mathcal{G}^{\prec} \leftarrow$ DetriangulateMoralGraph$(\hat{\mathcal{G}}, \Sigma, \sigma)$
$\mathcal{G}^{\prec} \leftarrow$ ConstraintPropagation$(\mathcal{G}^{\prec})$
DAGs $\leftarrow$ get equivalent graphs of $\mathcal{G}^{\prec}$
return DAGs
```

3.2.4. Maximal DAG to Equivalent Networks

The maximal DAG could possibly contain several equivalent graphs encoding the same joint probability distribution. Traditional structure learning methods will also end up with an equivalence class of networks. As a final step, it is worth converting the partial DAG (PDAG) to its equivalent completely oriented DAGs (CDAGs). This can be done using dynamic programming. Each undecided edge will be assigned an orientation manually, in the meantime, the constraints should always be updated and propagated. Thus it largely reduces the size of resulting networks and therefore is of great practical use. Finally, we present the PICM-CBN algorithm combining the Copula based parameter learning in Algorithm 3. We adopted the Gaussian Copula. Note that, in Gaussian Copula, the estimated parameter Σ can work as the input for the structure learning based on partial inverse correlation matrix. Thus we gain a significant improvement in running time. Even though there is a large number of missing observations, the learned parameters still suffice for the structure learning by applying a robust non-parametric density estimation on univariate marginals (Elidan, 2010b).

4. Experiments and Results

This section presents the evaluation of the proposed algorithm on first, synthetic and second, a real world data set. The experiments on the synthetic data sets address the scalability and parameter dependencies of the algorithm, while the second part on the real world data set compares the resulting network with prior knowledge in this domain and thus shows its applicability.

4.1. Synthetic Networks

The synthetic data sets comprise five synthetic networks with node size $5, 7, 10, 20, 50$ which were created and sampled with the Bayesian network toolbox (BNT) (Murphy). This was repeated 10 times to examine data variability effects. The size of the data sets (number of instances) ranges from 100 to 5000. None of these networks represents a model of any real-world domain, they are just created to represent causal relationships among variables.

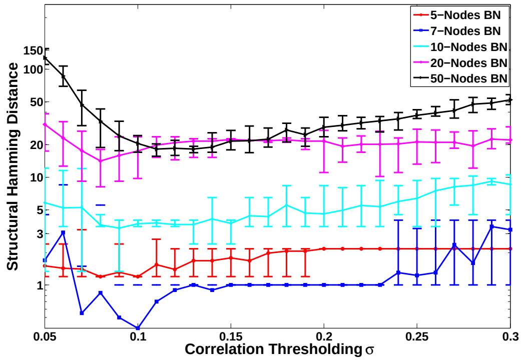


Figure 2. SHD against σ on five synthetic networks (node size 5, 7, 10, 20, 50, respectively) with training sample size = 1000 using PICM-CBN.

In order to compare the learned structures with the original networks, we use the Structural Hammming

Distance (SHD) (Acid, 2003), which calculates the minimal number of operations converting one network into another. One important parameter of the *PICM-CBN* algorithm is the correlation zero-approximation threshold σ (cf. Alg. 3) that controls the number of edges in the graph. However, it is not obvious how to fix that threshold appropriately. Therefore, the first evaluation examines the resulting networks by varying σ. Figure 2 shows that SHD approaches a minimum when σ is around 0.1, in the meantime, it achieves a stabler performance with less variance on SHD. For further experiments, σ will be fixed to the value of 0.1, which is assumed to be reasonable.

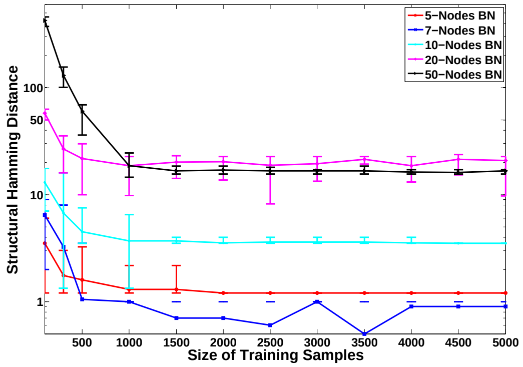


Figure 3. SHD against various training sample sizes on five synthetic networks (node size 5, 7, 10, 20, 50, respectively) using PICM-CBN ($\sigma = 0.1$). Sample size ranges from 100 to 5000.

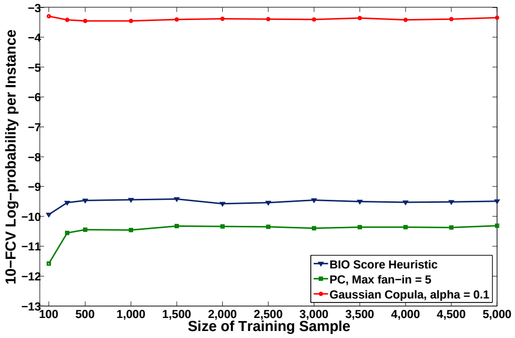


Figure 4. Average log-probability per instance on test-set against various training sample sizes (range from 100 to 5000) evaluated with 10FCV, on BIC scoring function based heuristic BN Learning, PC Algorithm where Max-fan-in = 5 and thresholding $\alpha = 0.05$, and also on PICM-CBN BN learning, where correlation thresholding $\sigma = 0.1$.

The next experiment addresses the algorithm's accuracy depending on the data set size. Figure 3 shows that the SHD decreases for larger data sets and reaches a promising value when the sample size exceeds 1000. Besides, for smaller sample size, the algorithm is not stable enough with high variances in SHD. Note that the SHD also correlates with the size of network, smaller networks are usually better identified than larger ones. In practice, a data set with a size above 500 should be considered as significant.

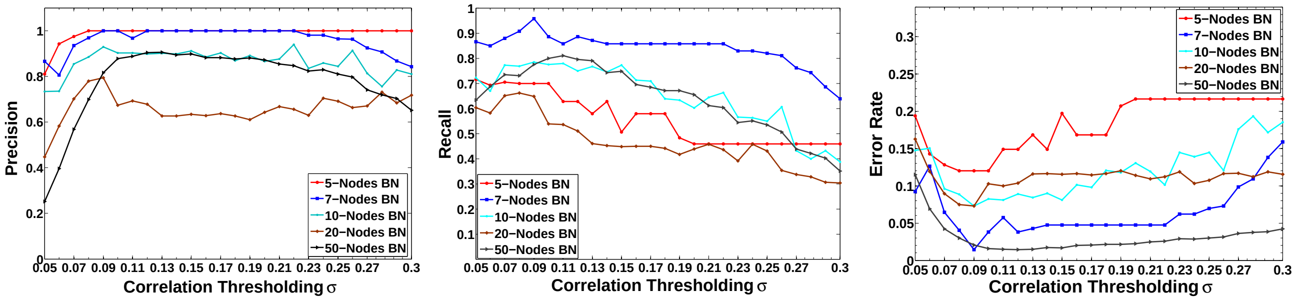
To evaluate the quality of the parameter learning, we ran experiments on data sets of various size using 10-fold cross-validation. We computed the average log-probability per instance over all test folds. Let lp denote the log-probability per instance in a test set D_t of size m, then: $lp = \frac{1}{|D_t|}\Sigma_{i=1}^{m} log(Pr(X_i))$, where X_i is the i-th instance in D_t. PC algorithm and BIC score function based method were taken for comparison. Both of these two algorithms adopt MLE as parameter estimation strategy. Since BIC score heuristic method is computationally quite demanding for learning large networks, this experiment was only run on the 5-Nodes network. It is obvious from Figure 4 that Copula based parameter estimation outperforms the other two on any training sample size.

Additionally, Figure 5 shows Precision, Recall and the error rate of SHD against various σ on the five synthetic data sets. Even the 50-Nodes network approaches a precision of 0.9 when σ is around 0.1. This means that 90% of the inferred edges appear indeed in the original network. The Recall values for all the learned networks except the 5-Nodes and 20-Nodes stay beyond 0.7 when the σ is around 0.1. This indicates that over 70% edges of original networks can be correctly inferred. The error rate represents the total accuracy of learned structures, mostly it is beneath 0.1, which corresponds to 90% accuracy.

The last experiment compares the *PICM-CBN* algorithm with the PC algorithm on the synthetic networks of size 5, 7, and 10. Figure 6 gives the runtimes and the SHD. PICM-CBN performs slightly better on structure learning than the PC algorithm for these small sized networks, and remarkably better in terms of runtime. Moreover, PICM-CBN is also more stable than PC, especially when the data set size is above 1000.

4.2. Real World Data Set

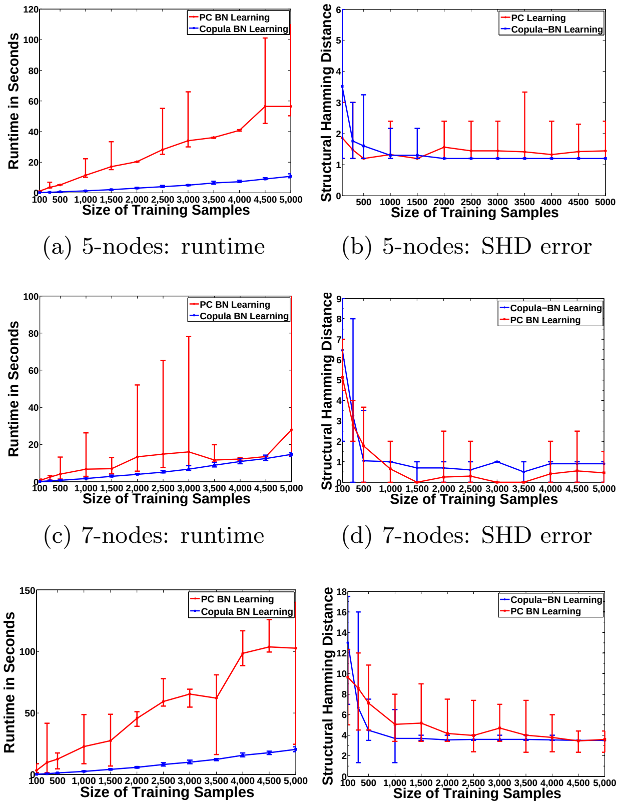
The second experiment part was run on the Abalone data set (Nash & Laboratories, 1994) taken from UCI repository (Frank & Asuncion, 2010). The Abalone data set contains 4177 samples and 9 attributes in total, of which there is one nominal attribute indicating the sex of abalone, which was removed for this experiment. The original task of this data set is to predict the age of an abalone, which is usually done by count-



(a) Precision (b) Recall (c) Error rate

Figure 5. Precision (a), Recall (b) and error rate (c) against various σ ranging from 0.05 to 0.3 on five synthetic networks (node size 5, 7, 10, 20, 50) with training sample size = 1000 using PICM-CBN.

ing the number of rings of the abalone. Again, the PC algorithm was taken for comparison. Figure 7 shows the log-probability of the induced networks for both algorithms.



(a) 5-nodes: runtime (b) 5-nodes: SHD error

(c) 7-nodes: runtime (d) 7-nodes: SHD error

(e) 10-nodes: runtime (f) 10-nodes: SHD error

Figure 6. Runtime and SHD error tests against various training sample sizes (range from 100 to 5000) on three synthetic networks (size 5,7,10), for PICM-CBN correlation thresholding $\sigma = 0.1$, for the PC algorithm max-fan-in = 5 and PC thresholding $\alpha = 0.05$.

This indicates that *PICM-CBN* outperforms the PC algorithm regarding both the predictive accuracy and stability. Moreover, the results on the abalone data set looks quite attractive with roughly an average log-probability of -1.7 per instance for both training and test data sets.

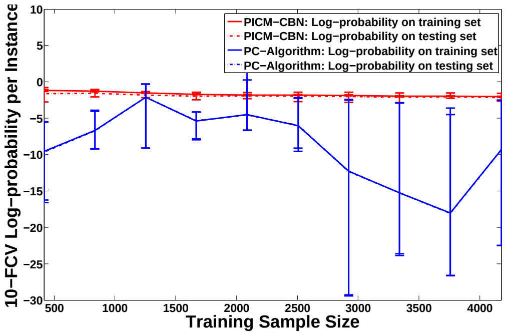


Figure 7. Log-likelihood for the training and test set for *PICM-CBN* and PC algorithms against the sample size on the Abalone data set (*PICM-CBN*: $\sigma = 0.1$, PC: max-fan-in $= 5, \alpha = 0.05$).

In order to learn the network structure, we run *PICM-CBN* multiple times on the full data set ($\sigma = 0.1$). The resulting network structure with corresponding variable names is shown in Figure 8. The learned network contains only 3 bidirectional edges and 12 edges in total. The variable *#rings* located in the center possesses the most connections which implies that the age of abalone is impacted by most of the other given variables. For example, *#rings* is directly influenced by the abalone's *weight*, *shucked weight* and *viscera weight*. The features regarding abalone's weight are highly connected, whereas the features about abalone's size including length and diameter are influenced by *#rings* (i.e., age). It is hard to assess

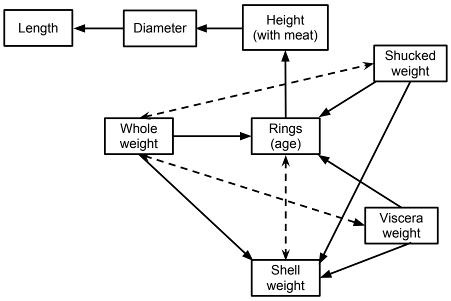


Figure 8. The learned network structure for Abalone data set.

the correctness of this causal model, but it corresponds quite well to prior knowledge about abalone. Therefore, this model can also be used to predict the age of abalone given the other variables.

5. Conclusion

In this paper, we presented a new method for the induction of Copula Bayesian Networks to model multivariate continuous distributions. We show that in Gaussian Copulas, instead of a large amount of independence tests, the PICM based structure learning method makes use of the estimated parameters of the Copula function to reduce the computational complexity. Experiments on the synthetic data sets show that our algorithm offers improvements on both structure learning and parameter learning. Moreover, comparisons to a BIC score heuristic-approach and the well known PC algorithm suggest that the PICM-CBN algorithm comes to better results in less time. Furthermore, we showed that the induced network on a real life data set also gives reasonable results. In the future, we first want to explore the impact of different families of Copulas on parameter estimation and structure learning. Second, we would like to investigate the use of PICM-CBN for the reconstruction of gene regulatory networks.

References

Acid, S.; de Campos, L. M. Searching for bayesian network structures in the space of restricted acyclic partially directed graphs. *Journal Of Artificial Intelligence Research*, 18:445–490, 2003.

Elidan, Gal. Copula bayesian networks. In *The 24th Annual Conference on Neural Information Processing Systems(NIPS)*, pp. 559–567, 2010a.

Elidan, Gal. Inference-less density estimation using copula bayesian networks. *Uncertainty in Artificial Intelligence (UAI)*, 26, 2010b.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Friedman, Nir. Learning Bayesian Network Structure from Massive Datasets: The ”Sparse Candidate” Algorithm. In *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence*, pp. 206–215, 1999.

Heckerman, David. Tutorial on learning in bayesian networks. Technical Report MSR-TR-95-06, Microsoft, 1995.

Korf, Richard E. Linear-space best-first search. *Artificial Intelligence*, 62(1):41–78, July 1993.

Murphy, Kevin. Bayes net toolbox for matlab. URL http://code.google.com/p/bnt/.

Nash, W.J. and Laboratories, Tasmania. Marine Research. *The Population biology of abalone (Haliotis species) in Tasmania: Blacklip abalone (H. rubra) from the north coast and the islands of Bass Strait.* Sea Fisheries Division, Dept. of Primary Industry and Fisheries, Tasmania, 1994.

Nelsen, Roger B. *An Introduction to Copulas.* Springer Science+Business Media, Inc., 2006.

Pearl, Judea. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, 2000.

Sklar, A. *Fonctions de repartition a n dimensions et leurs marges*, volume 8. Publications de lInstitut de Statistique de LUniversite de Paris, 1959.

Spirtes, P., Glymour, C., and Scheines, R. *Causation, Prediction, and Search.* The MIT Press, 2000.

Spirtes, Peter, Glymour, Clark, and Scheines, Richard. *Causation, Prediction, and Search.* The MIT Press, second edition, January 2001.

Whittaker, Joe. *Graphical Models in Applied Multivariate Statistics.* John Wiley & Sons, Ltd, 2008.