A Framework for Fault Diagnosis using Continuous Bayesian Network and Causal Inference

Asif Hanif
Electrical Engineering Dept.
Information Technology University
Lahore, Pakistan
asif.hanif@itu.edu.pk

Saad Ali
Electrical Engineering Dept.
Information Technology University
Lahore, Pakistan
saadali1906@gmail.com

Ali Ahmed
Electrical Engineering Dept.
Information Technology University
Lahore, Pakistan
ali.ahmed@itu.edu.pk

***Abstract*—Fault diagnosis in industrial facilities has traditionally been done using rule-based approaches, heuristics or expert-knowledge. Bayesian network provides a flexible and data-driven alternative that can reason under uncertainty. Most of the data being generated by sensors in industrial setups are continuous and the underlying data-generating models are essentially non-linear. This paper employs Bayesian network and proposes a framework that learns parameters of probability density functions of a continuous Bayesian network using neural network/s without requiring assumption of linear Gaussian model or discretization of continuous data. Moreover, an expression of probability query using learned parametric density functions and causal-inference based mathematical formulation of two tasks related to fault diagnosis –in the context of industrial plants– namely root-cause-analysis and identification of most-influential-path in Bayesian network have been provided.**

***Index Terms*—RCA, Bayesian network, parameter estimation, causal inference, neural network**

I. Introduction

Bayesian network (BN) – a type of probabilistic graphical model – is a simple and yet very powerful tool for reasoning under uncertainty. There are numerous practical applications of BN especially in genomics [1], software troubleshooting [2], prognosis and diagnosis of faults [3] and diseases [4] to name a few.

Our prime motivation for this work comes from root-cause-analysis (**RCA**) of faults and identification of most-influential-path (**MIP**) in industrial manufacturing plants. RCA is a process in which primary cause or reason of failure mode of an industrial plant is identified. The failure modes include, but not restricted to, equipment breakdowns in production-line, defects in the final product and lower quality of manufactured output. On a production-floor of an industrial plant, there may be multiple paths[1] from root-cause location to the point where failure is detected. To prioritize the debugging process, plant maintainers are usually interested in finding a path, from root-cause node to failure detection point, that has maximum contribution in creating faulty product i.e. MIP. Identification of MIP helps plant maintainers/technicians to fix the anomalous parts of production line, having highest contribution in creating defective output, with priority so that plant may be brought into normal operating condition as early as possible.

Traditionally, RCA and identification of MIP are performed by on-site engineers/technicians using expert knowledge. This manual procedure takes a lot of time in complex situations and is prone to mistakes due to experts' own biases. A more than expected downtime of plant can result in huge losses. For instance, "*one minute of downtime of an automotive manufacturing plant can incur USD 30,000 in operational expenses*" [5]. Secondly, it is humanly impossible to be aware of every process and take advantage of huge data, produced by sensors monitoring the industrial plant, by just visualization. Therefore, it is quintessential to have a data-driven algorithm that performs fully-automatic RCA, identifies MIP and provides *interpretable* insights in a minimum of time after fault detection.

We model machines/instruments (along with their inter-connections) of an industrial facility with a continuous[2] Bayesian network (CBN). To perform RCA and identify MIP in an industrial plant, we use synthetic data generated by nodes/sensors, structure of CBN and causal inference (details can be found in §II-D and §II-E).
After discovering the structure of a CBN, parameters of conditional probability density functions (PDFs) are estimated using an appropriate method. In general, parameter estimation of a non-linear[3] CBN is a challenging task. Most of the existing related literature tackles the problem of parameter estimation of CBN in two ways: (**i**) discretize the continuous data and convert CBN into discrete one, (**ii**) assume linear Gaussian model (LGM) for CBN [6]–[8]. There are certain disadvantages associated with these two approaches. Firstly, discretization of continuous data entails loss of information and requires fine bins/levels for *reliable* estimation of marginal/conditional distributions. Discretization with more number of bins/levels increases the size[4] of conditional probability tables(CPTs) which in turn increases the computational load on inference algorithm and slows down the decision making. Moreover,

[1] Here *path* refers to a particular sequence of machines/instruments in an assembly/production line of an industrial plant.

[2] In *continuous* Bayesian network, all nodes/random-variables are assumed to be continuous.

[3] *non-linear* relationship between a child node and its parents

[4] the number of parameters in marginal/conditional distributions

it requires huge amount of data to accurately estimate CPT of a child node having large number of parents. Secondly, LGM assumption is very restrictive and it fails to capture non-Gaussian distributions and non-linear relationship between a child node and its parents (which is often the case in real-world situations). Therefore, it is preferable to have a method for parameter estimation that directly operates on continuous data, learns parametric PDFs and does not require discretization or LGM assumption.

A Bayesian network factorizes joint distribution into product of uni-variate distributions which can later be used to compute different probability queries i.e. to perform probabilistic inference. In general, time complexity of exact inference methods in graphical models is NP-hard [9]. A general approach is to draw samples from a distribution and obtain an approximate answer of probability query [8]. Inference methods for a continuous and linear Gaussian Bayesian network are well established, however, a non-linear and non-Gaussian continuous Bayesian network poses challenges for inference [10]. There are a number multi-variate probability density functions for which there is no closed-form expression to evaluate high dimensional integrals , thus making exact inference difficult.

We assume a **non-linear** and **continuous** Bayesian network (CBN) with known structure and offer following contributions in this paper;

- Parameter learning in a CBN without either discretization of data or linear Gaussian assumption
- Adaptation of two simple methods for parameter learning in CBN with multiple fully-connected neural networks or single masked neural network
- A method for approximate inference involving learned multi-modal *logistic* PDFs and *Monte Carlo*
- Mathematical formulation and demonstration of RCA and identification of MIP in CBN using causal inference

Rest of the paper is organized as follows: §II-A gives introduction of Bayesian network, §II-B and §II-C mathematically formulate parameter learning and inference query respectively. Formal description of RCA and MIP using Bayesian network and causal-inference is given §II-D and §II-E respectively. Two simple methods to learn parameters of a CBN using neural network/s have been given in §III. A concise answer–derived using learned parametric PDFs and sampling method–of inference query can be found in §IV. Finally, results are presented in §V.

II. Preliminaries

A. Bayesian Network

A Bayesian network $\mathcal{G}(V, E)$ is a directed acyclic graph in which V is a set of nodes and E is a set of ordered pairs of nodes representing directed edges. Each node $v_i \in V$ is associated with a corresponding random variable X_i and $e_{i,j} = (v_i, v_j) \in E$ represents a directed edge from node v_i to node v_j i.e. $(v_i \rightarrow v_j)$. Assuming there are N continuous nodes in $\mathcal{G}$, the set of random variables associated with nodes is $\boldsymbol{X} = \{X_1, X_2, \dots, X_N\}$. We will use same notation for a node and the corresponding random variable i.e. X_i.

Let $f(X_1, X_2, \dots, X_N; \boldsymbol{\theta})$ be a joint probability density function (PDF) –parameterized by $\boldsymbol{\theta}$– of nodes/random-variables in $\mathcal{G}$ and assuming that joint PDF $f()$ is *Markov relative* [11] to $\mathcal{G}$, one can factorize joint PDF according to following equation;

$$f(X_1, X_2, \dots, X_N; \boldsymbol{\theta}) = \prod_{i=1}^{N} f\big(X_i \mid \mathrm{pa}(X_i)\ ; \boldsymbol{\theta}_i\big) \tag{1}$$

where $\mathrm{pa}(X_i)$ is a set[5] containing the parents of node X_i in $\mathcal{G}$. $f(X_i \mid \mathrm{pa}(X_i))$ is uni-variate conditional PDF of node X_i given its parents and it is parameterized by $\boldsymbol{\theta}_i$. $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^{N}$.

Assume D independent realizations of each node in $\mathcal{G}$, collectively denoted as $\mathcal{D} = \{\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \dots, \boldsymbol{X}^{(D)}\}$ where $\boldsymbol{X}^{(j)}$ denotes j_{th} sample vector containing values of all N nodes/random-variables i.e. $\boldsymbol{X}^{(j)} = [X_1^{(j)}, X_2^{(j)}, \dots, X_N^{(j)}]$. Maximum likelihood estimate (MLE) of Bayesian network $\mathcal{G}$'s parameters can be written as;

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \prod_{j=1}^{D} f(\boldsymbol{X}^{(j)}; \boldsymbol{\theta}) \tag{2}$$

where $\mathcal{L}$ is likelihood function, $\boldsymbol{\theta}$ contains parameters of $\mathcal{L}$ and $f()$ denotes joint probability density function over N nodes in $\mathcal{G}$. Using (1) and (2), one can show that;

$$\begin{aligned}\hat{\boldsymbol{\theta}} &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \sum_{i=1}^{N} \log \prod_{j=1}^{D} f\big(X_i^{(j)} \mid \mathrm{pa}(X_i^{(j)})\ ;\ \boldsymbol{\theta}_i\big) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \sum_{i=1}^{N} \log\ \mathcal{L}_i\big(\boldsymbol{\theta}_i\ ;\ X_i \mid \mathrm{pa}(X_i)\big)\end{aligned} \tag{3}$$

where $\mathcal{L}_i$ is local likelihood function associated with node X_i and $\mathrm{pa}(X_i^{(j)})$ denotes the values of parents of X_i in j_{th} observation/sample of $\mathcal{D}$. Equation (3) suggests that $\hat{\boldsymbol{\theta}}$ can be obtained by optimizing each local negative log-likelihood independently.

In this paper, we take mixture-of-logistic (MoL) for marginal and conditional PDFs as it is very flexible to represent wide range of distribution shapes (see Fig. 1) and has closed-form expression of cumulative distribution function (CDF) [12].

$$f\big(X_i \mid \mathrm{pa}(X_i); \boldsymbol{\theta}_i\big) = \sum_{m=1}^{M} \omega_{i[m]}\, \text{logistic}(\ \mu_{i[m]}\ ,\ s_{i[m]}\ ) \tag{4}$$

where $\boldsymbol{\theta}_i = \{\omega_{i[m]}, \mu_{i[m]}, s_{i[m]}\}_{m=1}^{M}$ and $(\omega_{i[m]}, \mu_{i[m]}, s_{i[m]})$ represent (weight, mean and scale) of m_{th} component of mixture. It must be noted that $\boldsymbol{\theta}_i$ is a function of the values of $\mathrm{pa}(X_i)$ and we have opted not to use superscript j with $\boldsymbol{\theta}_i$ for simplicity.

[5] Depending upon the context, it may also represent values assigned to the parents of node X_i

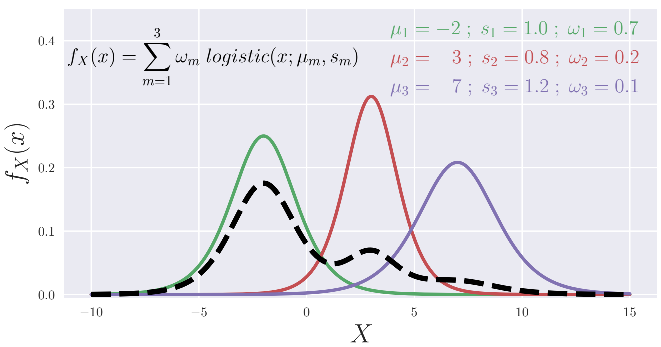


Fig. 1. A multi-modal logistic PDF (black dashed-line) can be obtained by taking convex combination of multiple uni-modal logistic PDFs. By varying parameters of individual components, shape of $f_X(x)$ can be changed accordingly. (Best viewed in color)

B. Parameter Learning

A Bayesian network is parameterized by $\{\boldsymbol{\theta}_i\}_{i=1}^{N}$. In case of discrete Bayesian network, these parameters are simply CPTs. However, for a CBN, CPTs are replaced by marginal/conditional PDFs.

For each continuous node X_i in $\mathcal{G}$, we aim to learn an arbitrary function ξ_i that takes in values of parents of node X_i and returns parameters $\boldsymbol{\theta}_i$ of conditional PDF $f(X_i \mid \text{pa}(X_i))$.

$$\xi_i : \quad \text{pa}(X_i) \rightarrow \boldsymbol{\theta}_i \tag{5}$$

For instance, ξ_i for an LGM will be a function that returns mean[6] and standard deviation. Since we assume MoL for $f(X_i \mid \text{pa}(X_i))$, therefore, ξ_i will map values in $\text{pa}(X_i)$ to $\{\omega_{i[m]}, \mu_{i[m]}, s_{i[m]}\}_{m=1}^{M}$ (see (4)). For a node X_i having no parents, i.e. root-node, the function ξ_i will take zero value as input and return parameters of marginal PDF of X_i, i.e. $\xi_i : \quad 0 \rightarrow \boldsymbol{\theta}_i$. In general, ξ_i could be any non-linear function, therefore, we use neural network to learn this function (see details in §III).

C. Inference

A parameterized Bayesian network can be used to compute probability queries of interest. Since we are dealing with continuous Bayesian network, therefore, probability query of the following form will be useful for most of the inference tasks.

$$\begin{aligned}&\Pr\Big[\mathit{event} \;\Big|\; \mathit{evidence}\Big] = \\ &\Pr\left[\bigcap_{i \in I}(x_i^{'} \leq X_i \leq x_i^{''}) \;\middle|\; \bigcap_{j \in J}(x_j^{'} \leq X_j \leq x_j^{''})\right]\end{aligned} \tag{6}$$

where Pr represents probability, I is a set denoting indices of nodes in event and J is a set denoting indices of nodes in condition/evidence of probability query. Moreover, $(x_i^{'}\;,\;x_i^{''})$ is an interval containing values of node X_i.

In §IV, we provide an expression to compute probability query (shown in (6)) using parametric PDFs. In addition to it, following two subsections give mathematical formulation of RCA and MIP using causal inference and CBN. These two problems are inspired by their application in *corrective maintenance* of industrial manufacturing facilities.

D. Root Cause Analysis (RCA)

As mentioned in §I, the goal of RCA is to identify primary cause or reason for a failure mode in a process. Many real-world processes can be modeled using Bayesian networks [13]. In the context of graph theory, a fault or undesirable condition is originally initiated by a node, named as root-cause or *source* node (X_s). Moreover, a node where fault is detected or observed is being referred to as *sink* node (X_t) here. Source and sink nodes can be anywhere in graph $\mathcal{G}$ and they do not necessarily have to be one of the *root* or *leaf* nodes respectively.

In general, a process remains in healthy condition for most of the time, providing many samples of *healthy data*. At the occurrence of fault, only a few samples are available in *faulty data*. For RCA, we first learn parameters of BN $\mathcal{G}$ with healthy data and then perform following inference task using samples from faulty data to identify *source* node.

$$X_s = \underset{X \in \boldsymbol{X} \setminus \{X_t\}}{\text{argmax}} \;\; \Pr[\text{ Fault} \mid do(X = x_f) \;] \tag{7}$$

where X is any node in Bayesian network except the *sink* node X_t where fault is defined/detected. x_f is the value of node X observed in faulty data. Definition of fault is based upon a particular range of node X_t values e.g. Fault : $X_t > 10$. $do()$ is a *do*-operator that severs the connections coming from parents of X towards itself in $\mathcal{G}$ and assigns a fixed value to X. $do(X = x_f)$ makes X independent of its parents and forces this node to operate in its faulty value. Intuitively, $\Pr[\text{ Fault} \mid do(X = x_f) \;]$ can be interpreted as probability of the occurrence of fault when node X is forced to operate in its faulty value. If this probability turns out to be very high, it is indicative of X having high contribution in the occurrence of fault.

E. Most Influential Path (MIP)

RCA gives industrial plant maintainers information about the primary source of fault i.e. X_s. Since there may be multiple directed paths between X_s and X_t, therefore, to prioritize corrective actions on the targeted sequence of nodes, one must quantify impact of each path (between X_s and X_t) on the creation of fault condition. Following inference task will return sorted list of paths along with their impact. A path with highest impact is declared as most-influential-path.

Assuming there are K directed paths from X_s to X_t in $\mathcal{G}$ i.e. $(\boldsymbol{p}_k\;,\; k \in \{1, 2, \cdots, K\})$ and let $\mathcal{G}_k$ be an adjusted/path-specific Bayesian network for path $\boldsymbol{p}_k$, MIP (denoted as $\boldsymbol{p}^*$) between X_s and X_t is found as

$$\boldsymbol{p}^* = \underset{\substack{\boldsymbol{p}_k \\ k \in \{1,2,\cdots,K\}}}{\text{argmax}} \;\; \Pr_{\mathcal{G}_k}[\text{ Fault} \mid do(X_s = x_f) \;] \tag{8}$$

whereas in this case x_f represents value of X_s in faulty data. $\Pr_{\mathcal{G}_k}[.]$ notation indicates that probability is being found with respect to path-specific Bayesian network $\mathcal{G}_k$ and it quantifies

[6] linear combination of the values of parents of node X_i

the effect of $do(X_s = x_f)$ –propagated via path $\boldsymbol{p}_k$– on the occurrence of fault. A path-specific Bayesian network $\mathcal{G}_k$ is obtained by making modifications in original BN $\mathcal{G}$ such that all paths between X_s and X_t (except $\boldsymbol{p}_k$) get *de-activated*. We adapt the method for the formation of path-specific Bayesian network from [14]. A more detailed version is provided in supplementary material (§S4 of [15]).

III. Parameter Learning with Neural Networks

Since a neural network acts as a universal function approximator [16], therefore, one can use it to learn an arbitrary function mentioned in (5). Following are two methods for parameter learning in a CBN using neural network/s.

A. Multiple Fully-Connected Neural Networks

In this mode of parameter learning, each node X_i in $\mathcal{G}$ will have its own fully-connected (FC) neural network, let's say $\mathcal{NN}_i$ (see Fig. 2(a)). Number of neurons or units in the input-layer of $\mathcal{NN}_i$ will be equal to the number of parents of node X_i in $\mathcal{G}$. For each root-node, marginal PDF is estimated and there will be only one unit (holding constant zero value) at input-layer of its neural network. Number of hidden layers and number of units in each hidden layer are hyper-parameters. Number of units in output layer of $\mathcal{NN}_i$ will be equal to the number of parameters of conditional PDF $f\big(X_i \mid \mathrm{pa}(X_i)\big)$ i.e. $\boldsymbol{\theta_i}$. Since we are using MoL for each PDF (see (4)), therefore, output layer will contain $M \times 3$ units where M is the number of components in the mixture of logistic PDFs. Loss function of $\mathcal{NN}_i$ will be local negative log-likelihood function associated with node X_i. Implementation of loss function for this method and the subsequent one is similar to that of used by [12], however, we do not use *edge cases* because support of random variables in our case is not confined to $[0, 255]$ and we do not expect *peaks* at the edges of support. A formal description about the implementation of loss function can be found in supplementary material (§S1 of [15]). Parameter estimation using multiple neural network has also been done by [17], however, there are a number of differences between our approach and that of [17], e.g. we use MoL for marginal/conditional PDFs which is much more expressive as compared to uni-modal Gaussian PDF.

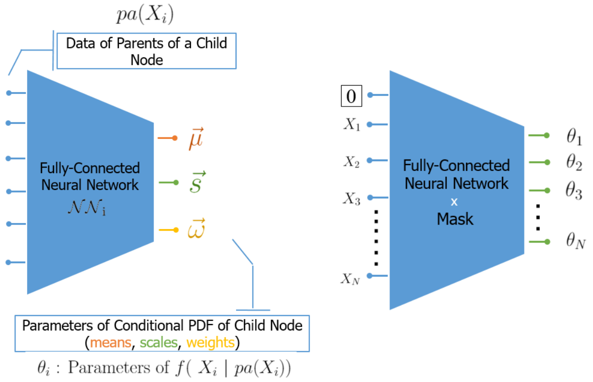
B. Single Masked Neural Network

One FC neural network can only be used for density estimation of one child node conditioned on its parents. Although separate fully-connected neural networks can be trained in parallel, however, one can also use a single masked-neural network to learn parameters of PDFs of all nodes in $\mathcal{G}$ [18]. It requires masking of fully-connected layers in such a way that only the parents of a particular child contribute in the estimation of its probability density parameters at output-layer.

For each layer of masked neural network;

$$\boldsymbol{y}_\ell = g\big((W_\ell \odot M_\ell)\boldsymbol{y}_{\ell-1} + \boldsymbol{b}_\ell\big)$$

where $\boldsymbol{y}_{\ell-1} \in \mathbb{R}^n$ and $\boldsymbol{y}_\ell \in \mathbb{R}^m$ are input and output of ℓ_{th} layer respectively. $W_\ell \in \mathbb{R}^{m \times n}$ is weight matrix, $M_\ell \in \{0,1\}^{m \times n}$ is binary mask and $\boldsymbol{b}_\ell \in \mathbb{R}^n$ is bias vector associated with ℓ_{th} layer. $\odot$ is hadamard or point-wise product and g is a non-linear function.


$pa(X_i)$
Data of Parents of a Child Node
Fully-Connected Neural Network
$\mathcal{NN}_\text{i}$
$\vec{\mu}$
$\vec{s}$
$\vec{\omega}$
Parameters of Conditional PDF of Child Node (means, scales, weights)
θ_i : Parameters of $f(\ X_i \mid pa(X_i))$
0
X_1
X_2
X_3
X_N
Fully-Connected Neural Network x Mask
θ_1
θ_2
θ_3
θ_N

(a) (b)

Fig. 2. (**a**) Graphical illustration of parameter learning using multiple separate fully-connected neural networks. Each node X_i will have its own neural network $\mathcal{NN}_i$. (**b**) Graphical illustration of parameter learning using single masked neural network.

To maintain dependence of child node only on its parents, each unit or neuron in neural network is labeled with a tag. Units in input and hidden layers are labeled with the names of nodes for which IsParent(X_i)=True in $\mathcal{G}$ and a *dummy* node which is assumed to be hypothetical parent of all root nodes in $\mathcal{G}$. Value of *dummy* node is always kept zero so that it may be mapped to parameters of marginal PDF of root-nodes (see (5)). Units in output layer are labeled with the names of all nodes in $\mathcal{G}$ i.e. $\{X_1, X_2, \cdots, X_N\}$.

After labeling of all units in neural network, binary masks are constructed using a method suggested by [18] with slight modifications. Weight matrix of each layer is multiplied by a binary mask. For hidden layers and output layer, masks are constructed according to (9) and (10) respectively. In the following equations, row index i corresponds to a unit in ℓ_{th} layer and column index j corresponds to a unit in $\ell - 1_{\text{th}}$ layer.

For hidden layers;

$$M_\ell[i,j] = \begin{cases} 1 & \text{label of } i_{\text{th}} \text{ unit is same as that of } j_{\text{th}} \text{ unit} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

For output layer;

$$M_\ell[i,j] = \begin{cases} 1 & \text{node corresponding to label of } i_{\text{th}} \text{ unit} \\ & \text{is child of } j_{\text{th}} \text{ unit} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

These masks ensure that parameters of conditional PDF of node X_i are only influenced by its parents $\mathrm{pa}(X_i)$. During training, data of the nodes– for which IsParent(X_i)=True in $\mathcal{G}$– are given at the input layer of masked-neural network. The output layer of masked-neural network provides parameters of PDFs $f\big(X_i \mid \mathrm{pa}(X_i)\ ;\ \boldsymbol{\theta_i}\big)\ \ \forall\ i \in nodes(\mathcal{G})$ (see Fig. 2(b)). In this case, loss function is the sum of all local negative log-likelihood functions (as shown in (3)).

IV. Inference using Learned Parametric PDFs

If a random variable X has logistic PDF i.e. $f(X = x) = \text{logistic}(x\ ; \mu, s)$, its CDF is $F_X(x) = \sigma((x - \mu)/s)$. One can easily compute probability of X falling in an interval as follows;

$$\begin{aligned}\Pr[x^{'} \leq X \leq x^{''}] &= \int_{x^{'}}^{x^{''}} f(X = x)\mathrm{d}x \\ &= F_{_X}(x^{''}) - F_{_X}(x^{'})\end{aligned}$$

Although there is a closed-form expression for CDF of univariate logistic PDF, however, there is no known analytical expression for CDF of multi-variate logistic PDF. Therefore, we adopt a hybrid technique incorporating parametric PDFs and *Monte Carlo* approximation to find the answer of probability query shown in (6).

For the following results, we assume that nodes $\boldsymbol{X} = \{X_1, X_2, \cdots, X_N\}$ in Bayesian network $\mathcal{G}$ are topologically sorted and follow breadth-first-search (BFS) ordering. Let p, q be the nodes at deepest level in $\mathcal{G}$ among the nodes indexed by $I \cup J$ and J respectively i.e. $p = \max(I \cup J)$ and $q = \max(J)$. Moreover, let $\mathcal{P}$ and $\mathcal{Q}$ be the regions in $\mathbb{R}^{p-1}$ and $\mathbb{R}^{q-1}$ defined by Cartesian product of intervals $\{(x_k^{'}\, x_k^{''})\}_{k=1}^{p-1}$ and $\{(x_k^{'}\, x_k^{''})\}_{k=1}^{q-1}$ respectively. It must be noted that if $k \notin I \cup J$, the corresponding interval $(x_k^{'}\, x_k^{''})$ is assumed to be $(-\infty, +\infty)$.

Using consequence of probability axioms and learned parametric conditional PDFs, following result is derived;

$$\Pr\left[\bigcap_{i \in I}(x_i^{'} \leq X_i \leq x_i^{''}) \,\middle|\, \bigcap_{j \in J}(x_j^{'} \leq X_j \leq x_j^{''})\right] = \\ \frac{\mathbf{E}_{f(\boldsymbol{X}_{<p})}\left[\mathbf{1}_{\mathcal{P}}(\boldsymbol{p}).\Pr\left[(x_p^{'} \leq X_p \leq x_p^{''}) \mid \text{pa}(X_p) = \pi_p(\boldsymbol{p})\right]\right]}{\mathbf{E}_{f(\boldsymbol{X}_{<q})}\left[\mathbf{1}_{\mathcal{Q}}(\boldsymbol{q}).\Pr\left[(x_q^{'} \leq X_q \leq x_q^{''}) \mid \text{pa}(X_q) = \pi_q(\boldsymbol{q})\right]\right]} \tag{11}$$

where $\boldsymbol{p} \in \mathbb{R}^{p-1}$, $\boldsymbol{q} \in \mathbb{R}^{q-1}$ are drawn from joint PDFs $f(\boldsymbol{X}_{<p}) = f(X_1, X_2, \cdots, X_{p-1})$ and $f(\boldsymbol{X}_{<q}) = f(X_1, X_2, \cdots, X_{q-1})$ respectively. $\pi_p()$ is a function that extracts values of node X_p's parents from input vector $\boldsymbol{p}$. Expectations in (11) can be approximated using *Monte Carlo* method. Detailed derivation of (11) is given in supplementary material (§S6 of [15]).

V. Results

For quantitative comparison of two uni-variate and multimodal *logistic* PDFs (true: f_X and predicted: $\hat{f}_X$), we use total-variation-distance (TVD). To calculate TVD, two PDFs are binned with L uniform intervals and area under each bin is computed using CDF– resulting in categorical distributions with L states i.e. $f_X \rightarrow P_X$ and $\hat{f}_X \rightarrow \hat{P}_X$. TVD between P_X and $\hat{P}_X$ is defined as ℓ_1 norm of their element-wise difference and it is bounded by closed interval $[0, 1]$.

$$TVD(\ P_X, \hat{P}_X\ ) = \frac{1}{2}\left\|P_X - \hat{P}_X\right\|_1$$

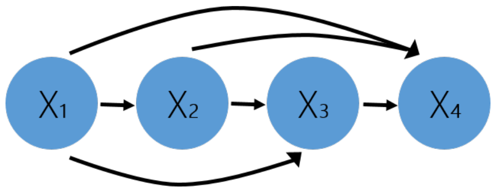


Fig. 3. A fully-connected Bayesian network over four *continuous* nodes.

A. Parameter Learning

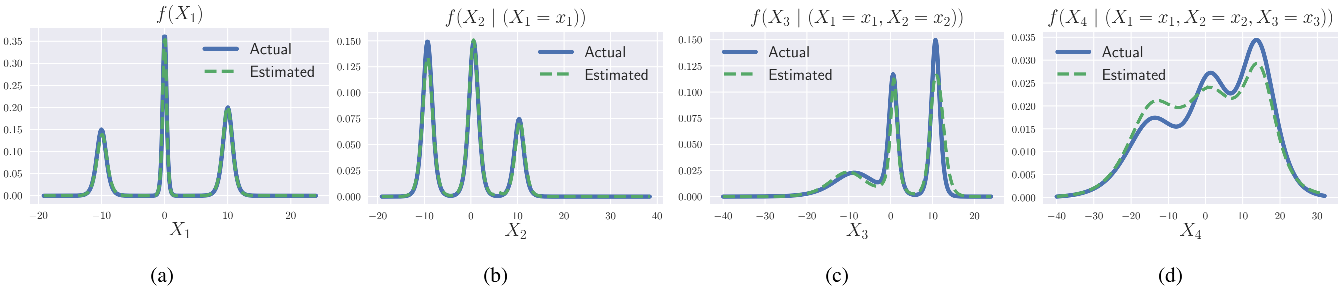
To present results of parameter learning, we use a Bayesian network shown in Fig. 3. It has four continuous nodes. Data generating process (from which train and test data are drawn) of this network is given in supplementary material (§S5.1 of [15]). Using the method described in §III-A, we used four separate neural networks– each having three fully-connected layers and each layer containing 28 units. We set the number of mixture components (i.e. M, see (4)) to 5 because we do not expect conditional PDFs to have more than 5 modes. During training, a neural network $\mathcal{NN}_i$ associated with node X_i takes the values of $\text{pa}(X_i)$ at input layer and outputs parameters of conditional PDF $f(X_i \mid \text{pa}(X_i))$. For instance, input layer of $\mathcal{NN}_3$ has two units for the values (X_1, X_2) and output layer has 15 units for parameters of $f(X_3 \mid (X_1 = x_1, X_2 = x_2))$ i.e. $\boldsymbol{\theta}_3 = \{\omega_{3}[m], \mu_{3}[m], s_{3}[m]\}_{m=1}^{5}$.

We draw 10,000 samples (of each node) from data generating process to train neural networks and use RMSprop optimizer with lr=0.01 and number of epochs set to 3000. Once training is complete, we predict conditional PDFs using test data (drawn from same data generating process). While generating test data, we also save parameters of conditional PDFs from which samples are drawn and use them to make comparison with the predicted ones.

Fig. 4 shows actual and predicted PDFs (obtained through separate trained neural networks). (a) is marginal PDF of root node X_1 and (b,c,d) show conditional PDFs of child nodes (X_2, X_3, X_4) respectively. For quantitative comparison, refer to Figs. [5, 6, 7]. Each individual violin-plot shows rotated kernel density plot of TVD between 50 true and predicted distributions. We denote approach used in this paper as **MoL** (mixture-of-logistic) and compare its results with the ones obtained from linear-Gaussian-model **LGM** and **Discretization**[7]. Fig. 5 shows TVD between true and predicted distributions of each node. It can be observed that TVD for MoL approach remains lower as compared to other two methods. Moreover, we also show the performance of these three methods w.r.t number of nodes in Bayesian network and number of samples in training data in Fig. 6 and Fig. 7 respectively. These results suggest that MoL approach has clear advantages over other two methods.

Parameter learning results obtained using single masked neural network are quite similar to the ones obtained via multiple separate fully-connected neural networks. Due to

[7] In this approach, data of all continuous nodes are converted into discrete data and then discrete Bayesian network is parameterized by CPTs.



(a)

(b)

(c)

(d)

Fig. 4. Actual and predicted PDFs associated with the nodes of Bayesian network shown in Fig. 3. Visual inspection indicates that predictions are quite close to actual parameters. For quantitative comparison, we use TVD between two distributions.

space constraint, we have shown results of masked neural network in supplementary material (§S7 of [15]).

For results shown in Fig. 8, a two-node continuous Bayesian network is assumed with X being the parent of node Y. Left scatter plot shows samples drawn from actual distribution. We use same MoL approach and LGM method on this data and draw samples from learned models.

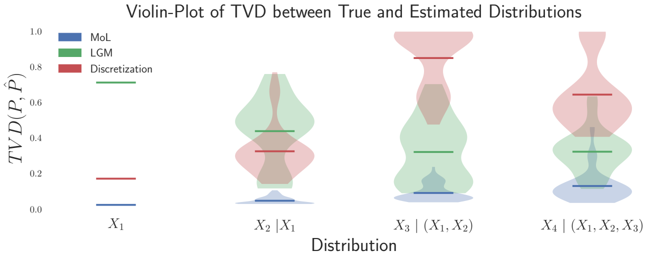


Fig. 5. Violin-plot of TVD vs. each node in Bayesian network. Horizontal bar represents mean value of corresponding kernel density plot.

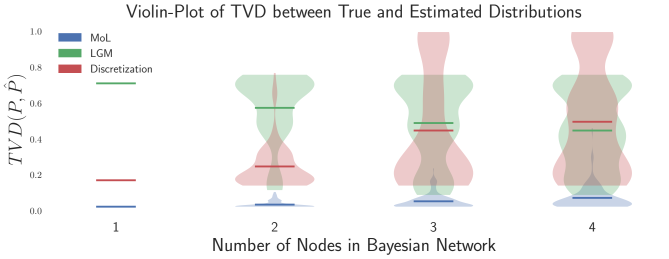


Fig. 6. Violin-plot of TVD vs. number of nodes in Bayesian network. Horizontal bar represents mean value of corresponding kernel density plot.

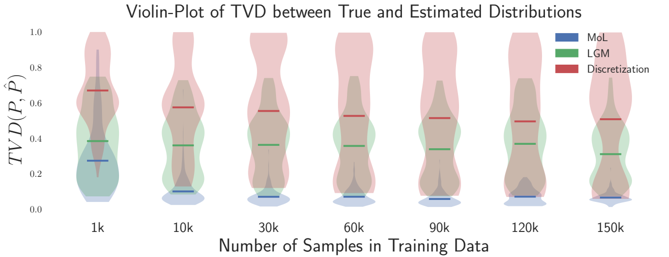


Fig. 7. Violin-plot of TVD vs. number of training samples. Horizontal bar represents mean value of corresponding kernel density plot.

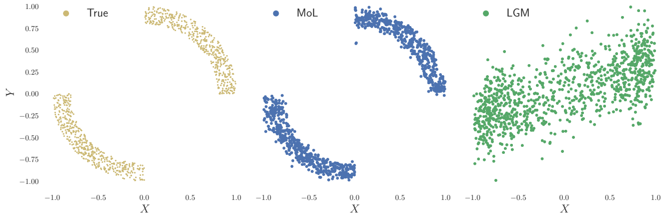


Fig. 8. For this graph, a two node Bayesian network $\{X \rightarrow Y\}$ is considered. Left scatter plot shows samples drawn from actual distribution. Middle and right plots show samples drawn using MoL approach and LGM respectively. LGM samples are quite different from the ones present in actual distribution.

B. Inference

In this section, we will show results of inference query (derived in §IV), RCA and MIP. Referring to Bayesian network shown in Fig. 3, we compute answers of nine different probability queries using expression shown in (11) and parametric PDFs obtained via trained neural networks. Since ground-truth is not available in the form of analytical closed-form expression, we compare our results with empirical[8] probabilities obtained using 5000 samples of test data. Fig. 9 shows empirical probabilities and estimated probabilities (computed according to (11)) of nine inference queries. Estimated probabilities agree with the empirical probabilities by an acceptable margin.

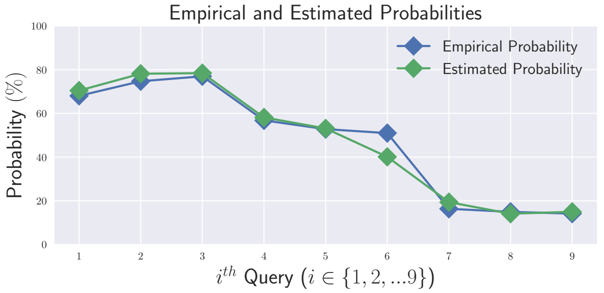


Fig. 9. Empirical and estimated probabilities of nine inference queries e.g. 9_{th} query is $\Pr[(-30 < X_4 < 1)\&(-15 < X_3 < 0) \mid (-15 < X_1 < 15)\&(-15 < X_2 < 15)]$.

Root Cause Analysis (RCA) and MIP

We demonstrate RCA and MIP using a thirteen node *continuous* Bayesian network shown in Fig. 10. Data generat-

[8] Empirical probability is obtained by $\frac{\text{No. of Samples in Event \& Evidence}}{\text{No. of Samples in Evidence}}$

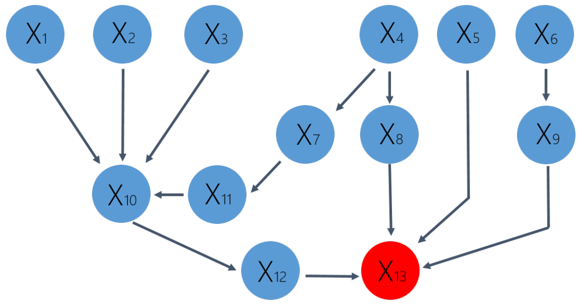


Fig. 10. A Bayesian network over thirteen *continuous* nodes. X_{13} is *sink* node where fault is defined/detected.

ing process of this network is given in supplementary material (§S5.2 of [15]). We define fault condition on node X_{13} and consider it to be *sink* node i.e. $X_t = X_{13}$. Fault is defined as an event in which the value of node X_{13} exceeds 120 i.e.

$$\text{Fault} : X_{13} > 120$$

1) Root Cause Analysis (RCA): For RCA, we generate healthy data (i.e. with no fault observed in data) and train 13 full-connected neural networks using the method described in §III-A with this healthy data. While generating faulty or test data (i.e. the samples in which value of node X_{13} exceeds 120), we perform an intervention on one of the nodes $\{X_1, X_2, \cdots, X_{12}\}$. Then we perform inference task mentioned in §II-D. We present RCA results for only two situations here: In first case, actual root-cause node is X_5 and in second case, actual root-cause node is X_4.

There are two possibilities for the root-cause node X_s:

(a) The X_s is directly connected to X_t in $\mathcal{G}$
(b) One or more nodes are present in the path/s originating from X_s and ending at X_t

In case **(a)**, only actual *source* node will have high interventional[9] probability and program in (7) will return correct root-cause node. For instance, Fig. 11 shows interventional probabilities when ground-truth root-cause node is X_5. As we can see that X_5 is the only one with significantly high interventional probability. If case **(b)** arises, node X_s and a few of its descendants may have high interventional probability and it is quite possible that interventional probabilities of some of these descendants are greater than that of actual root-cause node. To handle such case, one needs to list all the nodes whose interventional probabilities are greater than a certain threshold, explore their levels in the graph and then declare root-cause node that is at highest[10] level. We show results of such cases in supplementary material (§S3 of [15]).

Fig. 12 shows interventional probabilities when ground-truth root-cause node is X_4. It must be noted that descendants of X_4 are also having high interventional probability.

2) Most Influential Path (MIP): Once *source* node X_s has been determined using RCA, next step is to find MIP between X_s and X_t. For MIP, we take the case in which $X_s = X_4$.

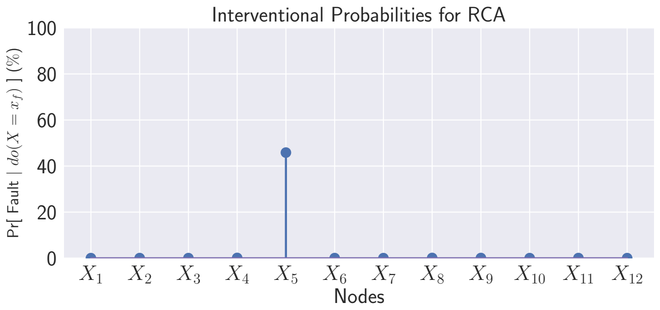


Fig. 11. RCA result for the case in which X_5 is the actual root-cause node which is directly connected to sink node X_t in Bayesian network shown in Fig. 10.

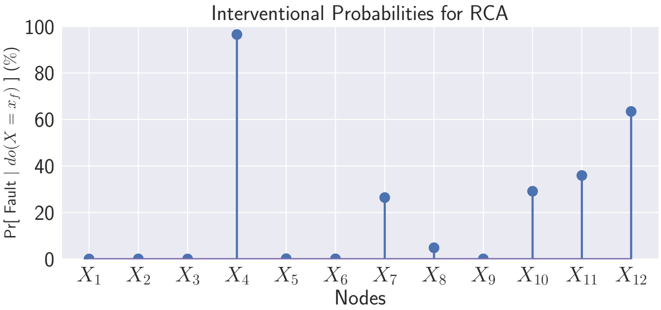


Fig. 12. RCA result for the case in which X_4 is the actual root cause. There are multiple nodes between X_4 and X_{13} in Bayesian network shown in Fig. 10.

There are two paths between X_s and X_t: $\boldsymbol{p}_1 = \{X_4 \rightarrow X_7 \rightarrow, X_{11} \rightarrow X_{10} \rightarrow X_{12} \rightarrow X_{13}\}$ and $\boldsymbol{p}_2 = \{X_4 \rightarrow X_8 \rightarrow X_{13}\}$. Since data are being generated synthetically, therefore, we set equations in such a way that contribution of path $\boldsymbol{p}_1$ remains higher as compared to that of $\boldsymbol{p}_2$. As mentioned in §II-E, we create an adjusted/path-specific Bayesian network $\mathcal{G}_k$ for each path and then find the probability of Fault condition under *do* operation on the value of X_4 taken from faulty data (see (8)). Resulting probabilities give us path-specific impacts on the fault. In this case, we get following probabilities:

$$\Pr_{\mathcal{G}_1}[\text{ Fault } | \ do(X_s = x_f) \ ] = 63 \ \%$$
$$\Pr_{\mathcal{G}_2}[\text{ Fault } | \ do(X_s = x_f) \ ] = 35 \ \%$$

Since $\Pr_{\mathcal{G}_1}[*] > \Pr_{\mathcal{G}_2}[*]$, therefore, $\boldsymbol{p}_1$ is declared as MIP which is according to our expectation. It must be noted that path-specific probabilities are calculated independently and they need not sum to one.

VI. Conclusion

We presented two simple methods for parameter learning in continuous Bayesian network using neural network/s. A method to compute probability query using learned parametric PDFs and *Monte Carlo* approximation is also presented. Moreover, mathematical formulation and demonstration of RCA and MIP on synthetic data (using continuous Bayesian network and causal inference) have been given. We believe that unified presentation of parameter learning using neural network/s and causal inference methods for fault diagnosis in this paper

[9] We refer probability of an event computed under $do()$ operation to as *interventional* probability.

[10] the closer to the root nodes, the higher the level

can help practitioners, especially the ones working on the *corrective maintenance* of industrial manufacturing plants.

References

[1] R. Agrahari, A. Foroushani, T. R. Docking, L. Chang, G. Duns, M. Hudoba, A. Karsan, and H. Zare, "Applications of bayesian network models in predicting types of hematological malignancies," *Scientific reports*, vol. 8, no. 1, pp. 1–12, 2018.

[2] J. Locke, "Basics of knowledge engineering," *Kindred Communications Troubleshooter Team, Microsoft Support Technology*, 2014.

[3] B. Cai, L. Huang, and M. Xie, "Bayesian networks in fault diagnosis," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2227–2240, 2017.

[4] M. Verduijn, N. Peek, P. M. Rosseel, E. de Jonge, and B. A. de Mol, "Prognostic bayesian networks: I: Rationale, learning procedure, and clinical use," *Journal of Biomedical Informatics*, vol. 40, no. 6, pp. 609–618, 2007.

[5] E. Josefsson, "Industry 4.0 cost of inaction and roi," tech. rep., Ericsson - ABI Research, 2019.

[6] Z. Ji, Q. Xia, and G. Meng, "A review of parameter learning methods in bayesian network," in *International Conference on Intelligent Computing*, pp. 3–12, Springer, 2015.

[7] N. Friedman, M. Goldszmidt, and T. J. Lee, "Bayesian network classification with continuous attributes: Getting the best of both discretization and parametric fitting.," in *ICML*, vol. 98, pp. 179–187, 1998.

[8] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[9] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[10] C. Li and S. Mahadevan, "Efficient approximate inference in bayesian networks with continuous variables," *Reliability Engineering & System Safety*, vol. 169, pp. 269–280, 2018.

[11] J. Pearl, *Causality*. Cambridge university press, 2009.

[12] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[13] J. Pearl and D. Mackenzie, *The book of why: the new science of cause and effect*. Basic books, 2018.

[14] J. Pearl, "Direct and indirect effects," *arXiv preprint arXiv:1301.2300*, 2013.

[15] A. Hanif, "Supplementary material: A framework for fault diagnosis using continuous bayesian network and causal inference." [Available] https://github.com/asif-hanif/supplementary_material/blob/main/Supplementary_Material.pdf.

[16] K. Hornik, M. Stinchcombe, H. White, *et al.*, "Multilayer feedforward networks are universal approximators.," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[17] S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien, "Gradient-based neural dag learning," *arXiv preprint arXiv:1906.02226*, 2019.

[18] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *International Conference on Machine Learning*, pp. 881–889, 2015.