Parameter learning algorithms for continuous model improvement using operational data

Madsen, Anders L.; Jeppesen, Nicolaj Søndberg; Jensen, Frank; Sayed, Mohamed S.; Moser, Ulrich; Neto, Luis; Reis, Joao; Lohse, Niels

Published in:
Symbolic and Quantitative Approaches to Reasoning with Uncertainty

DOI (link to publication from Publisher):
10.1007/978-3-319-61581-3_11

Publication date:
2017

Document Version
Accepted author manuscript, peer reviewed version

Parameter Learning Algorithms for Continuous Model Improvement Using Operational Data

Anders L. Madsen[1,2]✉, Nicolaj Søndberg Jeppesen[1], Frank Jensen[1], Mohamed S. Sayed[3], Ulrich Moser[4], Luis Neto[5], Joao Reis[5], and Niels Lohse[3]

[1] HUGIN EXPERT A/S, Aalborg, Denmark
[2] Department of Computer Science, Aalborg University, Aalborg, Denmark
[3] Loughborough University, Loughborough, United Kingdom
[4] IEF-Werner GmbH, Furtwangen, Germany
[5] Instituto de Sistemas e. Robotica Associacao, Porto, Portugal

Abstract. In this paper, we consider the application of object-oriented Bayesian networks to failure diagnostics in manufacturing systems and continuous model improvement based on operational data. The analysis is based on an object-oriented Bayesian network developed for failure diagnostics of a one-dimensional pick-and-place industrial robot developed by IEF-Werner GmbH. We consider four learning algorithms (batch Expectation-Maximization (EM), incremental EM, Online EM and fractional updating) for parameter updating in the object-oriented Bayesian network using a real operational dataset. Also, we evaluate the performance of the considered algorithms on a dataset generated from the model to determine which algorithm is best suited for recovering the underlying generating distribution. The object-oriented Bayesian network has been integrated into both the control software of the robot as well as into a software architecture that supports diagnostic and prognostic capabilities of devices in manufacturing systems. We evaluate the time performance of the architecture to determine the feasibility of on-line learning from operational data using each of the four algorithms.

Keywords: Bayesian networks, parameter update, practical application

1 Introduction

The need for diagnostic and health monitoring capabilities in manufacturing systems is becoming increasingly important as manufacturing organisations continuously aim to reduce system downtime and unpredicted disturbances to production. We have found that Bayesian networks (BNs) [17, 3, 6] and their extension Object-Oriented Bayesian Networks (OOBNs) [8, 13] are well-suited to capture and represent uncertainty in root-cause analysis using both component-level models and wider system-level models integrating component-level models. The crucial need for diagnostic and health monitoring capabilities is accompanied with the availability of increasing amounts of sensory data and decreasing costs of computation on the shop-floor level have opened new opportunities for

component suppliers and system integrators to provide more competitive functionalities that go beyond traditional control and process monitoring capabilities.

In this paper, we consider the challenge of parameter learning for continuous model improvement using operational data. In particular, we investigate the use of four different approaches to improve the diagnostic performance of an OOBN using operational data. The four algorithms are the batch EM algorithm, incremental EM, Online EM and fractional updating. The investigation is performed using an OOBN for root-cause analysis of a pick-and-place industrial robot developed by IEF-Werner GmbH[6] (the Linear Axis shown in the center of Figure 3). An initial OOBN for root-cause analysis has been developed based on expert knowledge [11]. The OOBN has been integrated into the control software of the component and is being deployed in a production line where efficient and effective root-cause analysis is required in case of failure. In order to improve the diagnostic performance of the OOBN different methods for continuous model update based on operational data are being investigated. This paper reports on the results of these investigations.

Inspired by the work of [18], a number of approaches are considered. Notice that our work differs from the work of [18] in three important ways: (1) we are considering parameter learning in OOBNs, (2) the objective is to improve diagnostic performance (not classification), and (3) while [18] compares three algorithms, we investigate four algorithms. We consider the EM algorithm [9] for parameter learning from a batch of data (referred to as batch EM). Using batch EM, the idea is to collect data in batches and learn parameters off-line, for instance, during maintenance hours as suggested by [18]. We use batch EM as a reference. Adaptive causal probabilistic networks and fractional updating are described in [16] who cites [21] while adaptive probabilistic networks are described in [19] and [1]. A similar gradient descent approach is described in [5]. [10] describes how the approach of [16] referred to as sequential learning has been implemented in the HUGIN tool. The online EM algorithm [2] is a stochastic gradient method that is faster than other gradient methods such as [19] which involves a difficult task of determining the step size between iterations.

2 Preliminaries and Notation

A BN $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$ consists of a directed, acyclic graph G specifying dependence and independence relations over a set of variables $\mathcal{X}$ and a set of conditional probability distributions (CPDs) $\mathcal{P}$ encoding the strengths of the dependence relations effectively combining elements of probability and graph theory. A BN is a representation of a joint probability distribution $P(\mathcal{X}) = P(X_1, \ldots, X_n) = \prod_{X_i \in \mathcal{X}} P(X_i|\pi_{X_i})$ where π_X are the parents of X in G. The CPD $P(X|\pi_X)$ consists of one probability distribution over the states of X for each configuration of π_X. We only consider discrete variables.

An OOBN is a BN augmented with network classes, class instances and an associated notion of interface and private variables [8, 13, 6]. A class instance is

[6] http://www.ief-werner.de

the instantiation of a network class representing a sub-network within another network class. The variables $\mathcal{X}(C)$ of network class C are divided into disjoint subsets of input $\mathcal{I}$, output $\mathcal{O}$ and hidden $\mathcal{H}$ variables such that $\mathcal{X}(C) = \mathcal{I} \cup \mathcal{O} \cup \mathcal{H}$ where the interface variables $\mathcal{I} \cup \mathcal{O}$ are used to link nested class instances, see Figure 1. Inference in an OOBN is performed by creating a *run-time* instance of the model and doing inference in this model. A run-time instance of an OOBN is created by expanding it into a corresponding flat BN.

The Hellinger distance $D_H(P, Q)$ used to compare two probability distributions P and Q is defined as $D_H(P, Q) = \sqrt{\sum_i (\sqrt{p_i} - \sqrt{q_i})^2}$ [18] who cites [7]. It is similar to the Kullback-Leibler divergence, but defined for zero probabilities. To compare the results of parameter learning using two different algorithms on the same OOBN, the distance is computed as a sum of $D_H(P_i, Q_i)$ for $X_i \in \mathcal{X}$. This is similar to the approach taken by [22] and [18]. For each parent configuration π of each X in each network class C, $D_H(P_1(X|\pi), P_2(X|\pi))$ is computed where P_1 and P_2 are CPDs produced by the two learning algorithms. The values $D_H(P_1(X|\pi), P_2(X|\pi))$ are summed across parent configurations, variables and classes (ignoring bounded input nodes). In the weighted Hellinger distance $D_H^w(P_1(X|\pi), P_2(X|\pi))$, $D_H(P_1(X|\pi), P_2(X|\pi))$ is weighted by $P(\pi)$ in the reference model.

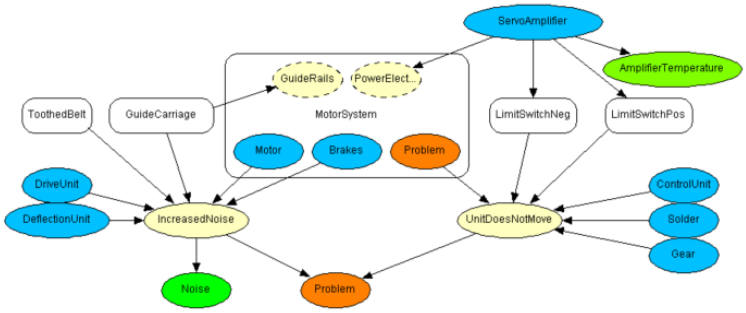

Fig. 1. The top level class of the Linear Axis Model.

3 The Linear Axis OOBN Model

The Linear Axis as a self-sustainable handling system that is designed to be a high performance machine with a demand to work 24h / day seven days a week. Therefore, there is little or no time for maintenance and repair. This means that there is a need for system condition monitoring to prevent failures and for system failure diagnosis. The Linear Axis diagnosis model considered here is used for root-cause analysis under the assumption that a problem is observed and the five most likely root causes should be identified. Figure 1 shows the structure of the top-level class of the Linear Axis OOBN. In the figure, blue nodes denote possible root causes, orange nodes denote problem defining nodes, and green nodes denote

possible observations such as sensor readings and operator feedback. The model has 35 variables, 27 failure states, 555 CPD entries, maximum CPD size of 128 and five class instances (two instances of the LimitSwitch class). The Linear Axis OOBN has been quantified using subject matter expert knowledge. We refer to this model as the knowledge driven model and its development is described in more detail in [11].

The diagnostic performance of the knowledge driven model has been assessed following the approach of [11]. The basic idea, is to iterate through the root causes where each root causes is instantiated to a failure state and all other root causes are instantiated to non-failure. For each such configuration, values for the observations are generated. The values for the observations are propagated in the model and the probabilities of the root causes recorded. This demonstrates how well the observations can distinguish the root causes.

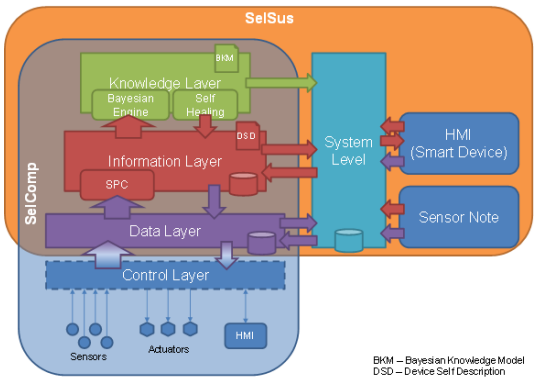

Fig. 2. The SelComp internal architecture concept.

4 The SelSus Architecture

The aim of the SelSus System Architecture is to provide an environment for highly effective, self-healing production resources and systems to maximize their performance over longer life times through highly targeted and timely repair, renovation and upgrading [20]. The architecture defines three levels of abstraction for its constituents: 1) Component Level, which relates directly to machines or its sub-components and is composed of smart sensory capabilities, methods for self-diagnostics and predictive maintenance. 2) Station Level, at this level the developments are constituted by previous capabilities plus human machine interfaces and tools to support the design and maintenance of the factory station. 3) Factory Level, previous levels capabilities are combined to create a semantic driven maintenance scheduling for large production factory plants.

The Linear Axis typically integrates a production cell, performing operations in collaboration with other machines (e.g., robotic arms and welding tools). To

make operational and sensory data available to the SelSus System, the SelComp (SelSus Component) concept was designed. The SelComp (Figure 2), is a self-aware entity that makes available to the SelSus system its internal state conditions, providing this way operational and structural knowledge. A SelComp also provides built-in models for state estimation based on sensor data which enables pro-active and predictive maintenance. These components have the ability to collect data from sensors that are mounted physically in the same device or in a near location and fuse this data to extend its models capabilities. The Linear Axis OOBN has been encapsulated in the Machine SelComp for the Linear Axis [20, 12], see Figure 3, as it represents a field device, machine or its sub-components. The goal is to provide diagnostic capabilities at component-level supporting system-level diagnostics. A Sensor SelComp [14, 15], on the other hand, is designed to provide essentially smart sensor data to the SelSus system and more often to Machine SelComps. The Sensor SelComp component has *plug&play* capabilities in terms of physical sensors, data models and algorithms. The Linear Axis OOBN can also be abstracted as a service to provide outputs to and subscribe to inputs from the SelSus System and other SelComps.

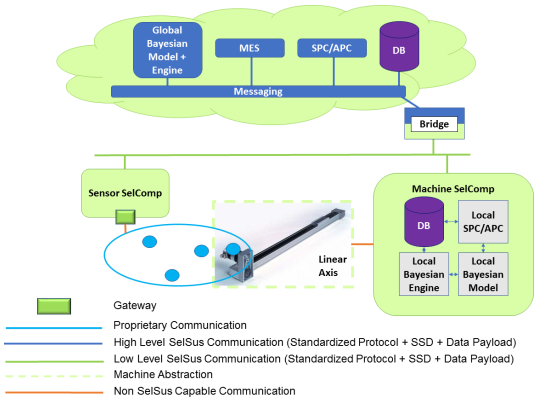

Fig. 3. The SelSus System Architecture.

5 Parameter Learning Algorithms

Let $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$ be a BN with parameters $\Theta = (\theta_{ijk})$ such that $\theta_{ijk} = P(X_i = k \mid \pi_{X_i} = j)$ for each i, j, k. The task of parameter learning is to estimate the values of the parameters Θ given a dataset of cases $\mathcal{D} = \{c_1, \ldots, c_N\}$. The cases of $\mathcal{D}$ are assumed independent and identically distributed (i.i.d.) with values missing at random or completely at random [4]. The generating probability distribution is assumed to be stationary. We first present the parameter learning algorithms for standard BNs followed by a description of how they are applied to OOBNs.

The EM algorithm [9, 4] is well-suited for calculating maximum likelihood and maximum a posteriori (MAP) estimates in the case of missing data. The algorithm iterates the E-Step and M-Step until convergence. Given an initial value of the parameters Θ, the E-step computes the current expected sufficient statistics while the subsequent M-step maximizes the log-likelihood function $l(\Theta \mid D) = \sum_{i=1}^{N} \log P(c_i \mid \Theta)$. The E-step of the EM algorithm computes expected sufficient statistics for each family $\mathrm{fa}(X_i)$ and parent configuration π_{ij} of each X_i under Θ as $n(Y) = \mathbb{E}_{\Theta}\{n(Y) \mid D\}$, where $n(\cdot)$ is counts for either π_{ij} or $X_i = k, \pi_{ij}$. The M-step computes new estimates of θ^*_{ijk} from the expected counts under θ_{ijk} and a virtual count $\theta_{ijk}\alpha_{ij}$ specified beforehand (MAP estimate):

$$\theta^*_{ijk} = \frac{n(X_i = k, \pi_{ij}) + \theta_{ijk}\alpha_{ij}}{n(\pi_{ij}) + \alpha_{ij}}, \tag{1}$$

where α_{ij} is the equivalent sample size (ESS) specified for π_{ij}.

The principle idea of the incremental EM algorithm, e.g. [18], is to divide the data $\mathcal{D}$ into disjoint subsets $\mathcal{D}_1, \ldots, \mathcal{D}_m$ and iteratively apply EM on $\mathcal{D}_i$. The estimates θ_{ijk} and α_{ij} produced by one iteration of EM are used as virtual counts in the next iteration of EM. If $\mathcal{D}$ is complete, then incremental EM and batch EM produce the same result. Incremental EM is less space demanding than EM as it only needs to hold $\mathcal{D}_i$ in memory at step i.

Online EM [2], which can be considered a gradient ascent algorithm, performs a parameter update after propagating each case c. It is a stochastic approximation algorithm that computes the updated parameter θ^*_{ijk} as:

$$\theta^*_{ijk} = (1 - \gamma)m_{ijk} + \gamma P(x_{ijk}|c), \tag{2}$$

where m_{ijk} is the normalized sufficient statistics computed as $m_{ijk} = \alpha_{ij} * p(x_{ik}|\pi_{ij}) / \sum_j \alpha_{ij}$ and $p(x_{ijk}|c)$ is computed by propagating case c. Notice that even for π_{ij} with $P(\pi_{ij}) = 0$ there is a fading of $(1 - \gamma)$. The learning rate $\gamma = (1 + n)^{-\rho}$ controls the weighting of new cases where n is the iteration number. [2] suggests to use $\rho = 0.6$ while [18] recommends $\rho = 0.501$.

The fractional updating algorithm, see e.g., [16] who cites [21], also performs a parameter update after propagating each case c. In fractional updating the parameter θ_{ijk} is adjusted after propagating each case c as follows:

$$\theta^*_{ijk} = \frac{\alpha_{ijk} + P(x_{ijk}|c)}{\alpha_{ij} + P(\pi_{ij}|c)}. \tag{3}$$

Fading of past cases is controlled by a fading factor λ specified for each π_{ij} and a gradual fading is obtained by taking $P(\pi_{ij})$ into consideration. To improve performance, fractional updating is only performed for π_{ij} when $P(\pi_{ij}) > 0$ (an update would leave θ^*_{ijk} and α_{ij} unchanged when $P(\pi_{ij}) = 0$)), see [10].

Both fractional updating and Online EM perform no update when $\alpha_{ij} = 0$. Also, fractional updating and Online EM are the least space demanding algorithms as they only need to hold the latest case in memory.

In the general case of OOBNs, we compute the average expected counts for the run-time instances of the node and increase the experience counts by the number of run-time instances. This applies to all four algorithms.

6 Empirical Evaluation

The empirical evaluation is organised into three different tests (1) we consider updating the parameters of the knowledge driven model where all distributions are made uniform using a dataset of 250,000 cases with 5% missing values generated completely at random from the knowledge driven model, (2) we consider updating the parameters of the knowledge driven model using an real operational dataset, and (3) we consider the time performance of updating the parameters in the knowledge driven model in a real setting. The evaluations are performed using different values of the parameters of the learning algorithms.

Figure 4 shows the results (1) where a random sample generated from the knowledge driven model is used to learn the parameters in the model with uniform distributions. Figure 4 (left) shows the weighted Hellinger distance while (right) shows the time usage of each algorithm.

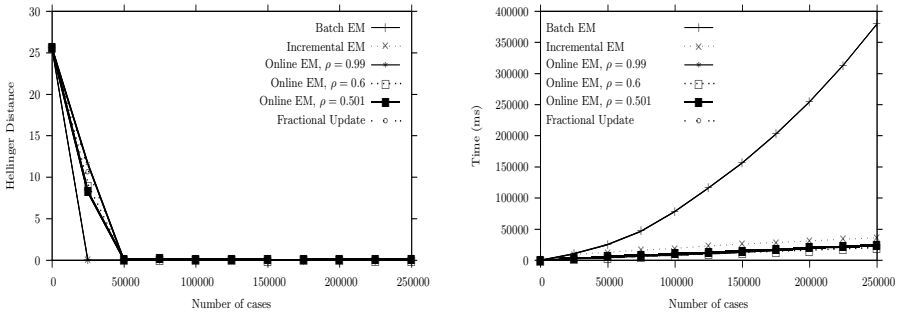

Fig. 4. Hellinger Distance (left) and accumulated time is ms. (right).

In the test, the values $\rho \in \{0.501, 0.6, 0.99\}$, $n_0 = 1$, $\sum_j \alpha_{ij} = 1$ for all i and $\lambda = 1$ are used. The value for $N = 0$ is the distance is between the uniform distribution and the knowledge driven model. The distances are quickly reduced in all cases and after a certain point no or little improvement is observed. D_H^w reduces the impact of large differences in distributions for π_{ij} with $\alpha_{ij} \ll 1$.

In (2) the operational dataset contains two sequences of 13,429 cases in total with six observed sensor readings represented in the model, i.e., there is 83% missing values due to the hidden variables alone. It contains both failure and non-failure cases. Table 1 shows the diagnostic performance of the five models considered where μ_{rank} refers to the average rank of the *true* root cause, i.e., the value 1 means perfect performance and 27 worst possible performance. In the test, the values $\rho = 0.99$, $\sum_j \alpha_{ij} = 13,429$ for all i and $n_0 = 13,429$ are used.

For all algorithms, there is an increase in the number of true root causes identified as the cause with highest probability. For Top-5 there is a significant improvement using fractional updating. The value μ_{rank} is not improved for

Table 1. The diagnostic performance of the five models considered.

Algorithm	Top-1	Top-5	μ_{rank}
Knowledge driven model	8	17	4.6
Batch EM	10	17	5.1
Online EM	9	17	4.5
Fractional update	10	21	3.4

batch EM. This is due to three true root causes obtaining a significantly worse rank after learning (e.g., rank 2 before compared to rank 16 after learning).

Next (3), we report on a performance analysis of two levels of integration of the OOBN model into the SelSus architecture using Online EM and fractional updating for parameter learning. The first and most tight level of integration has been achieved by integrating the model directly into the component control software where data is read from file. The second configuration is to deploy a BBN web service holding the model, a data server holding the data and the control software inside the SelSus Cloud. The control software retrieves data from the cloud and requests propagation of and learning from each case in the data retrieved. We consider retrieving different amounts of data in each request.

Table 2. Average time cost of handling one case across the integration levels.

Algorithm	Configuration	cases/request	Total time (ms)	Average time (ms)
Online EM	Direct integration	1	1,730	0.067
	SelSus Cloud	1000	11,367	0.44
	SelSus Cloud	100	44,867	1.74
	SelSus Cloud	10	496,199	19.29
Fractional	Direct integration		1,533	0.067
Updating	SelSus Cloud	1000	10,553	0.41
	SelSus Cloud	100	42,111	1.64
	SelSus Cloud	10	478,612	18.60

Table 2 shows the total and average time cost for each configuration. The analysis is performed using an operational data set of $25,726$ cases collected randomly. Here the focus is only on runtime and not the learning. As expected, there is a significant difference between direct integration and using a cloud service wrt. runtime. This means that the learning must be designed taking the data frequency into consideration.

7 Discussion

We have described the use of batch EM, incremental EM, Online EM and fractional updating on OOBNs for continuous model improvement using operational data. The objective was to improve the diagnostic performance of an OOBN for root-cause analysis by adjusting the model parameters using data.

The experimental results show that parameter learning for continuous model improvement using operation data is both feasible and will lead to better diagnostic performance. The Online EM algorithm is sensitive to the value of the ρ parameter and our results indicate that a value close to 1 is preferred. This is contrary to the [18] who suggests $\rho = 0.501$ and [2] who suggests $\rho \in [0.6; 0.9]$ and uses $\rho = 0.6$. The ρ-value controls the learning rate γ and higher γ-values (and lower ρ-values) means more emphasis on new cases (i.e., faster learning).

The results of the first experiment demonstrate that all four approaches quickly produce a model that has a low D_H^w relative to the knowledge driven models. There is no method that produce a significantly better result that the other algorithms. This is using a data set with 5% missing values. The running time of incremental EM with large batches and batch EM makes them infeasible in practice for this application.

For the Linear Axis OOBN with $||\mathcal{X}|| = 35$ only six are observed in the operational data. This data set has not been augmented with information on presence or absence of root causes nor any operator feedback. The data only includes sensor readings. Despite this fact, the algorithms all improve the diagnostic performance of the model compared to the initial knowledge driven model. It is expected that enriching the operational data with information on absence or presence of root causes will improve the learning. This will reduce the bias of the data as much more non-failure than failure data must be expected. Fractional update enables the specification of different α_{ij} for different CPDs and parent configurations in this way controlling the impact of the expert assessed parameters whereas Online EM uses normalised sufficient statistics and uses ρ to control the learning rate.

When learning the parameters from operational data using a knowledge driven model as the starting point, a decision on the relative balance of the data and the expert elicited values must be made. In the experiments, we have defined an ESS equal to the size of the operational data. This decision is important when the data stream is, in principle, infinite in a real operational setting. In any case, it is important that the parameters are stable at least until sufficient data has been processed, which is dependent on the model complexity.

Acknowledgments

This work is part of the project "Health Monitoring and Life-Long Capability Management for SELf-SUStaining Manufacturing Systems (SelSus)" which is funded by the Commission of the European Communities under the 7th Framework Programme, Grant agreement no: 609382. We would like to thank Andres Masegosa for discussions on the Online EM algorithm and the reviewers for their insightful comments, which have helped to improve the paper.

References

1. J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive Probabilistic Networks with Hidden Variables. *Machine Learning*, 29(2):213–244, 1997.

2. O. Cappe and E. Moulines. Online EM Algorithm for Latent Data Models. *J. Royal Statistical Society Series B (Statistical Methodology)*, 71(3):593–613, 2009.
3. R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.
4. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society - Series B*, B39(1):1–38, 1977.
5. F. V. Jensen. Gradient Descent Training of Bayesian Networks. In *Proc. ECSQARU*, pages 190–200, 1999.
6. U. B. Kjærulff and A. L. Madsen. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, 2nd edition, 2013.
7. G. Kokolakis and P. Nanopoulos. Bayesian multivariate micro-aggregation under the Hellingers distance criterion. *Research in Official Statistics*, 4(1):117–126, 2001.
8. D. Koller and A. Pfeffer. Object-Oriented Bayesian Networks. In *Proc. UAI*, pages 302–313, 1997.
9. S. L. Lauritzen. The EM Algorithm for Graphical Association Models with Missing Data. *Computational Statistics & Analysis*, 19:191–201, 1995.
10. A. L. Madsen, M. Lang, U. B. Kjærulff, and F. Jensen. The Hugin Tool for Learning Bayesian Networks. In *Proc. ECSQARU*, pages 594–605, 2003.
11. A. L. Madsen, N. Søndberg-Jeppesen, N. Lohse, and M. Sayed. A Methodology for Developing Local Smart Diagnostic Models Using Expert Knowledge. In *IEEE INDIN*, pages 1682–1687, 2015.
12. A. L. Madsen, N. Søndberg-Jeppesen, M. S. Sayed, M. Peschl, and N. Lohse. Applying Object-Oriented Bayesian Networks for Smart Diagnosis and Health Monitoring at both Component and Factory Level. In *Accepted for IEA / AIE 2017*, 2017.
13. M. Neil, N. Fenton, and L. M. Nielsen. Building large-scale Bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, 2000.
14. L. Neto, J. Reis, D. Guimaraes, and G. Concalves. Sensor cloud: Smartcomponent framework for reconfigurable diagnostics in intelligent manufacturing environments. In *IEEE INDIN*, pages 1706–1711, 2015.
15. L. Neto, J. Reis, R. Silva, and G. Concalves. Sensor SelComp, a Smart Component for the Industrial Sensor Cloud of the future. In *IEEE ICIT*, pages 1256–1261, 2017.
16. K. G. Olesen, S. L. Lauritzen, and F. V. Jensen. aHUGIN: A System Creating Adaptive Causal Probabilistic Networks. In *Proc. UAI*, pages 223–229, 1992.
17. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.
18. P. Ratnapinda and M.J. druzdzel. Learning discrete Bayesian network parameters from continuous data streams: What is the best strategy? *J. Applied Logic*, 13:628–642, 2015.
19. S. Russell, J. Binder, D. Koller, and K. Kanazawa. Local Learning in Probabilistic Networks With Hidden Variables. In *"Proc. of IJCAI"*, pages 1146–1152, 1995.
20. M. S. Sayed, N. Lohse, N. Søndberg-Jeppesen, and A. L. Madsen. SelSus: Towards A Reference Architecture for Diagnostics and Predictive Maintenance Using Smart Manufacturing Devices. In *IEEE INDIN*, page 6, 2015.
21. D. M. Titterington. Updating a diagnostic system using unconfirmed cases. *Applied Statistics*, 25:238–47, 1976.
22. A. Zagorecki, M. Voortman, and M.J. Druzdzel. Decomposing Local Probability Distributions in Bayesian Networks for Improved Inference and Parameter Learning. In *Proc. FLAIRS*, pages 860–865, 2006.