Rapidly Learning Bayesian Networks for Complex System Diagnosis: A Reinforcement Learning Directed Greedy Search Approach

WENFENG ZHANG, WENQUAN FENG, HONGBO ZHAO, AND QI ZHAO
Department of Electronic and Information Engineering, Beihang University, Beijing 100191, China
Corresponding author: Hongbo Zhao (bhzhb@buaa.edu.cn)

ABSTRACT Bayesian networks are a popular diagnosis method, whose structures are usually defined by human experts and parameters are learned from data. For the increasing complexity of modern systems, building their structures based on physical behaviors is becoming a difficult task. However, the improvement of data collection techniques motivates learning their structures from data, where greedy search is a typical iterative method. In each iteration, it generates multiple structure candidates by modifying one edge, evaluates these structures by scores based on data and selects the best structure for the next iteration. This method is costly because there are too many structures to be evaluated. To solve this problem, we frame the traditional greedy search by Markov decision process and propose an efficient Bayesian network learning approach by integrating reinforcement learning into it. In our approach, a convolutional neural network is employed as the value function to approximate scores. Before evaluating structures using data, the neural network is used to predict scores. Structure candidates with a low predicted score are discarded. By avoiding unnecessary computation, the cooperation of reinforcement learning and greedy search effectively improves the learning efficiency. Two systems, a 10-tank system with 21 monitored variables and the classic Tennessee Eastman process with 52 variables, are employed to demonstrate our approach. The experiment results indicate that the computation cost of our method was reduced by 30%–50%, and the diagnosis accuracy was almost the same.

INDEX TERMS Data driven fault diagnosis, Bayesian network, greedy search, reinforcement learning.

I. INTRODUCTION

For the development of sensor and data storage techniques, data-driven diagnosis (DDD) [1]–[6] received much attention recently. As one of the most popular data-driven methods, Bayesian networks (BNs) are widely applied to fault diagnosis [7]–[12].

Cai et al. [7] employed BNs to fuse multi-source information, increasing the diagnosis accuracy of ground-source heat pump (GSHP) system. They established the BN structure according to the cause and effect sequence of faults and symptoms, and studied the parameters using Noisy-OR and Noisy-MAX model. By combining two proposed BNs, their multi-source information fusion based fault diagnosis model increased the fault diagnostic accuracy greatly.

Mack et al. [8] learned BN structures to augment aircraft diagnostic reference model, which was a bipartite graph between condition indicators (CIs) and faults. They aimed to augment the original reference model by analyzing historical data. Tree augmented naïve Bayesian networks (TANs) were learned to derive diagnostic relations over CIs in the system. Then the reference model was updated by analyzing the learned TAN. Their method is useful not only for finding additional relations and monitors to augment subsystem reference models, but also provide significant indicators to knowledge engineers and system experts.

Codetta-Raiteri and Portinale [9] exploited the modeling features and inference capabilities of dynamic BNs (DBNs) in fault detection, identification, and recovery (FDIR) for autonomous spacecrafts. An FDIR cycle, which includes the tasks of diagnosis, prognosis and recovery, is introduced and characterized through a DBN model. Simulated problems showed that the DBN method was able to properly detect and deal with faults.

Zhao et al. [10] proposed an intelligent fault detection and diagnosis (FDD) strategy based on BNs for chillers.
Their BNs was composed of three layers: causal factors in layer 1, faults in layer 2 and fault symptoms in layer 3. In the diagnosis of refrigerant flow air conditioning systems, Hu et al. [12] developed a three-layer BN model as well: faults in the first layer, features in the second layer, and additional information in the third layer. In the two researches, posterior probabilities of faults under measurements were calculated for fault diagnosis. Compared with other fault diagnosis approaches, the two methods are able to utilize both system information and expert knowledge. More review can be found in the work of Cai et al. [11].

In most applications, including the approaches discussed above, the structures of BNs are established by human experts and the parameters are learned from data. However, building structures for complex systems is intractable for the number of possible structures is exponential on variable number [13].

TAN is one of the methods to learn both the structure and parameters. However, it only gives a tree structure which is oversimplified in complex cases. Learning Bayesian networks in a more general way is necessary for complex system diagnosis. Typically, Bayesian structure learning approaches can be classified into two categories: constraint- and score-based methods [13]. The first one builds the network skeleton based on independence test and then orients the skeleton. The second method, Greedy search (GS) for instance [13], employs a score to evaluate networks based on data and utilizes optimization algorithms to achieve a good structure. Because testing all possible conditional independences is intractable, most current researches focus on score-based approach.

Mohammadi and Wit [14] proposed a birth-death Markov Chain Monte Carlo (MCMC) method in Bayesian structure learning for sparse Gaussian graphical models. O’Gorman et al. [15] adopted quantum annealing in the search process. Haffa et al. [16] used regular vines to search sparse models. Tsamardinos et al. [17] combined constraint- and score-based method, where the skeleton was first obtained by constraint-based method and further optimized by a score-based method. Instead of traditional search method, Bartlett and Cussens [18] utilized integer linear programming (ILP) to solve the NP-hard optimization problem.

All the discussed methods suffer efficiency problem. Most of them can only make use of the current best solution(s). For instance, GS only modifies one edge of the current best solution to generate structure candidates in each iteration [13]. However, all evaluations in the search/optimization process contain structure information about the system. Extracting local structure information from the evaluation history can help to focus on the most likely structures. In this work, we implement this idea by reinforcement learning (RL) [19]–[21]. The score of structures is defined as the total rewards in RL, which is approximated by a convolutional neural network (CNN). Before computing the scores of new structures based on data in each iteration, the value function is employed to estimate the scores first. Only several structures with top estimated scores are further evaluated by data. In turn, the evaluated values are fed back to train the CNN. The combination of GS and RL significantly improves the learning efficiency.

The rest of the paper is organized as follows. Section 2 gives some basic concepts on BNs, the greedy search based BN learning algorithm, and reinforcement learning. Our method is elaborated in section 3, including how to frame the GS in a Markov decision process (MDP), the design of the value function and the details of the structure learning. We demonstrate it with a 10-tank system in the 4th part. Section 5 analyzes and discusses our approach. Conclusions are made in the final part.

II. BACKGROUND

As a probability graphical model (PGM), BNs are used to structurally represent complex probability distributions. This section introduces Bayesian network diagnosers, the greedy search based learning method and reinforcement learning.

A. BAYESIAN NETWORK DIAGNOSERS

Definition 1: Given variable set \( V \), a Bayesian Network is composed of two parts \((G, P)\) where \( G \) is the network structure represented by a directed acyclic graph (DAG) including all the variables in \( V \), and \( P \) are the parameters of conditional probability distributions (CPDs). In most cases, the CPDs are represented by conditional probability tables (CPTs). If there exist continuous variables, we have to discretize them [22].

B. GREEDY SEARCH BASED BAYESIAN NETWORK LEARNING

Score-based BN learning methods need a score to evaluate BNs. This paper adopts Bayesian information criterion (BIC) [13], [23], as shown in (1).

\[
\text{score}_{BIC}(G : D) = \log \hat{\theta} : D) - \frac{\log M}{2} \text{Dim}[G].
\] (1)

where \( \log \hat{\theta} : D) \) is the log likelihood between \( G \) and the historical data, \( \text{Dim}[G] \) represents the structural complexity and \( M \) is the size of the historical data. The log likelihood is given by (2) and \( \text{Dim}[G] \) is obtained by (3) and (4) [23] which is the number of independent parameters in \( G \).

\[
\log \hat{\theta} : D) = M \sum_{j=0}^{n-1} I_p(v_j|Pa(v_j)) - M \sum_{j=0}^{n-1} H_p(x_j)
\] (2)

\[
\text{Dim}[G] = \sum_{j=0}^{n-1} \text{Dim}[v_j, Pa(v_j)],
\] (3)

\[
\text{Dim}[v_j, Pa(v_j)] = (|v_j| - 1) \prod_{v\in Pa(v_j)} |v|,
\] (4)

where \( n \) is the number of variables, \( Pa(v_j) \) is the parent of \( v_j \), \( I_p(v_j|Pa(v_j)) \) is the mutual information between \( v_j \) and \( Pa(v_j) \), \( H_p(x_j) \) is the entropy of \( x_j \), and \( |v| \) is the size of variable \( v_j \). The pseudo code of greedy search is given in II-C. The algorithm iterates \( k \) times to find a good solution with an
initial structure $G_0$, where $G_0$ is usually an empty graph. In each iteration, the algorithm checks any two nodes. If there is no edge between the two nodes, edges (in two directions) are added to obtain new structures. If there is an edge, the edge is deleted or reversed to get new structures. All the new structures are evaluated. The best one in them is selected for the next iteration. The best structure in the final iteration is returned by the algorithm.

Algorithm 1 Greedy Search Based BN Learning

\begin{enumerate}
  \item iterate $k$ times:
  \item $G_{\text{set}} \leftarrow \{G_0\}$
  \item for any node pair $(i, j)$: $i > j$
  \item add, delete or reverse edge $i$-$j$ in $G_0$, insert new acyclic structures into $G_{\text{set}}$
  \item compute BIC scores for all $G$ in $G_{\text{set}}$
  \item $G_0 \leftarrow G$ with the maximal score in $G_{\text{set}}$
  \item return $G_0$
\end{enumerate}

C. REINFORCEMENT LEARNING

Reinforcement learning [19]–[21] is the area of machine learning that deals with sequential decision-making. As shown in Figure 1, for time-step $t$, the agent makes a decision, $a_t \in A$, based on observation $s_t$. Environment state transits from $s_t$ to $s_{t+1}$, and outputs observation $\omega_{t+1}$ and reward $r_t$ to the agent. RL aims to maximize the total rewards in the future. The value of a state is the maximum reward an agent can expect from the state in the future as shown by (5).

$$V(s) = \max_{a \in A}(E(r(s, a, s') + \gamma V(s'))), \quad (5)$$

where $\gamma$ is the discount factor that weights the future rewards. The value function (v-function) with $\gamma = 1$ is employed in this paper.

III. METHODOLOGY

This elaborates our approach. First, the greedy search based Bayesian network learning method is represented in the view of Markovian decision process, which is the base of RL. This part clarifies the motivation of combing GS and RL. Following that, the framework of our method is given, and each component is explained one by one.

A. GREEDY SEARCH: A MARKOV DECISION PROCESS

The idea to combine greedy search and reinforcement learning is inspired by their implicit common point: Markov decision process (MDP) [20], [21].

Definition 2: A discrete time stochastic control process is Markovian (i.e. it has the Markov Property) if

- $P(\omega_{t+1} | \omega_t, a_t) = P(\omega_{t+1} | a_t, \omega_t)$, and
- $P(r_t | \omega_t, a_t) = P(r_t | a_t, \omega_t)$

The Markov property means that the future of the process only depends on the current observation, and the agent has no interest in looking at the full history.

Definition 3: A Markov decision process is a 5-tuple defined by (6)

$$(S, A, T, R, \gamma), \quad (6)$$

where $S$ is the state space, $A$ is the action space, $T: S \times A \rightarrow S$ is the transition function, $R: S \times A \times S \rightarrow [0, R_{\text{max}}]$ is the reward function and $\gamma$ is the discount factor. If the observation in a system is the same with the state, $\omega_t = s_t$, the MDP is fully observable. As shown in Figure 2, in a fully observable MDP, an action is offered by a policy, the state transits into a new state based on the action, where the distribution of the new state is given by the transition function, and the reward is given by the reward function.

In our greedy search based BN learning, if we define BN structures as states, the iteration process is Markovian because for time-step $t$, only state $s_t$ is used to find a better solution. The actions defined by GS are to delete, add or reverse an edge. Its policy is to try all possible actions, obtain and evaluate all new states/structures, and finally select the best one. The transition is deterministic, and the reward is the increment of the score with discount factor $\gamma = 1$.

$$a_{t+1} = \max_{a \in A}\{\text{score}_{\text{BIC}}(s^a_{t+1} : D)\}, \quad (7)$$

$$r_t = \text{score}_{\text{BIC}}(s^a_{t+1} : D) - \text{score}_{\text{BIC}}(s_t^a : D), \quad (8)$$

$$V(s_t) = \text{score}_{\text{BIC}}(s_t : D). \quad (9)$$

Equation (7), (8) and (9) formally give the policy, reward function and value function of greedy search respectively, where $s^a_{t+1}$ is the structure at time-step $t+1$ if action $a$ is employed. Because equation (7) needs to evaluate all available structures based on data, the policy is inefficient.

Value-function based reinforcement learning methods solve an MDP in a more elegant way: using a function to approximate the value function and adopting the best estimated action in most cases.
The former discussion indicates that the key difference between greedy search and value-function based reinforcement learning is how to choose the action: greedy search tries all possible actions and selects the best one, while the value-function based reinforcement learning just selects the best one estimated by the value function.

The key shortcoming of the greedy search is the evaluation of all available new states, which can be expensive and unnecessary. We solve the deficiency by the value function in reinforcement learning, which helps to ignore structures with a low estimated score.

### B. REINFORCEMENT LEARNING DIRECTED GREEDY SEARCH

Our basic idea is to avoid evaluating unlikely BN structures by RL, as demonstrated in Figure 3. In the greedy search module, available structures are generated based on the current best one. Then if the value-function in the reinforcement learning module is accurate enough, the score of these structures is estimated by it, and only likely structures remain. Otherwise, all available structures are likely. The likely structures are further evaluated by data to generate the new best structure and the evaluated scores are fed back to the reinforcement learning module for the value-function training. The rest of this subsection elaborates the details of the framework.

1) THE REPRESENTATION OF BN STRUCTURES

As the definitions in the second section, a BN is composed of two parts: the structure and the parameter. This paper employs adjacent matrix to represent the structures of BNs. The structure of a BN is represented by a matrix $G$, where entry $G_{ij} \in \{0, 1\}$ ($i \neq j$). $G_{ij} = 1$ means that edge $i \rightarrow j$ exists and $G_{ij} = 0$ means that $i \rightarrow j$ does not exist. $G_{ij} \times G_{ji} \neq 1$.

In our diagnosis problem, in addition to the monitored variables, we employ another variable, $m$, to represent the health state/mode of systems, as illustrated in Figure 4.

In the search process, including both the classic greedy search and the proposed one, we start from a Naïve Bayesian network $G_0$, assume that edges from the mode variable to all other variables always exist and there is no edge directed to the mode variable. Learning algorithms only concerns the edges between different monitored variables.

2) CNN-BASED VALUE FUNCTIONS

There are multiple alternatives to approximate the value function, such as linear regression and polynomial functions [24]. This work employs CNNs which have proven to be powerful in image recognition [25], [26]. For their significant ability in feature extraction, CNNs are widely used in many fields, including fault diagnosis [27]–[33].

A CNN for image recognition is usually composed of multiple stacked layers. These layers can be typically categorized as convolutional layer, pooling layer and SoftMax layer [27].

Convolutional layers are employed to extract features from data and generate high level features. Convolutional kernels slide over data matrixes to extract local features, as illustrated in Figure 5 [34]. A kernel acts like a filter which is sensitive to a specific feature.

The learned features are de-sampled by pooling layers so that we do not have to handle too many features. Finally, the features with the highest level are merged by SoftMax layers to predict labels. (Sometimes, fully-connected layers are used before SoftMax layers as well).

The input of CNNs, images, are expressed in matrix. As BN structures are in the form of matrix as well [13], we can easily use CNNs for structure estimation. In practice, most monitored variables in fault diagnosis are only related to part of all the variables, and local features exist in BN structures. Considering that CNNs extract local features [34] from
images, it is feasible to utilize CNNs as the value-function for structure estimation. When the node order changes, local features change as well but a fixed order means some specified local features which can always be identified by kernels. Because we need to predict a real value instead of a label, the SoftMax layer is not used.

Our value function approximators accept one and output a real number, where the matrix is similar to channels in image recognition [26] and the real number is the estimated value. Figure 6 illustrates the structure of a value function approximator. The next subsection elaborates how to integrate GS and RL for BN learning. To be brief, g is employed to represent our CNNs.

3) REINFORCEMENT LEARNING DIRECTED GREEDY SEARCH

At the beginning of the greedy search, the value-function cannot be used to discard unlikely structures because it is not accurate enough, which is similar to the “exploration & exploitation” problem in RL. We solve it by introducing a piecewise function, as shown in Figure 7, where the horizontal axis is the iteration number and the vertical axis represents the number of new likely structures (to be evaluated). In addition to the current best structure, \((N + 1)\) structures are evaluated by data in each iteration.

In the first several iterations, the value function approximator is not accurate. So, it is advisable to evaluate as many structures as possible to collect materials for the approximator training. When the training loss of the value-function is small enough, the unlikely structure can be effectively recognized by the value-function. Then, we reduce the number of likely structures linearly till a minimal number. The function curve in Figure 7 is formally described by equation (10) and (11).

\[
N = \begin{cases} 
N_0, & k \leq k_0 \\
[a(k - k_0) + N_0], & k_0 < k \leq k_1 \\
N_1, & k_1 < k,
\end{cases} \tag{10}
\]

\[
\alpha = \frac{N_1 - N_0}{k_1 - k_0}, \tag{11}
\]

where \(k\) represents iteration index, \(N\) is the number of structures to be evaluated in each interaction, \(N = N_0\) in the first \(k_0\) interactions, \(N = N_1\) after the search iterates \(k_1\) times, and \(N\) linearly decreases and keeps an integer between \(k_0\) and \(k_1\).

A system with \(n\) monitored variables has \(n(n-1)/2\) unordered variable pairs, \(i - j\) (\(i > j\)). For a pair \(i - j\) in a BN structure, only one of the following three cases is true: no edge between \(i\) and \(j\), edge \(i \rightarrow j\), or edge \(j \rightarrow i\). We can 1) add edge \(i \rightarrow j\) or \(i \rightarrow j\) in the first case, and 2) delete or reverse the edge in the other two cases. So, by applying \(add\), \(delete\) and \(reverse\) operations once to any original BN structure, there are at most \(n(n-1)\) new structures. And we set \(N_0 = n(n-1)\) in our experiments.

Algorithm 2 Reinforcement Learning Directed Greedy Search

\[
\text{RLDGS}(G_0, D, g) \\
1. \text{score}_\text{set} \leftarrow \{\} \\
2. \text{iterate} \text{k} \text{times:} \\
3. \quad N \leftarrow \text{obtain the number of structures to be evaluated} \\
4. \quad G_\text{set} \leftarrow \text{obtain all possible new structures by applying operations once to } G_0 \\
5. \quad G_\text{set} \leftarrow \{G_0\} \cup \{\text{top } N \text{ acyclic structures estimated by } g \} \\
6. \quad D_\text{batch} \leftarrow \text{sample } D \\
7. \quad \text{evaluate structures in } G_\text{set} \text{ based on } D_\text{batch}, \text{ and add structure-score pairs into } \text{score}_\text{set} \\
8. \quad G_0 \leftarrow \text{the best structure in the last step} \\
9. \quad \text{optimize value function approximator } g \text{ using } \text{score}_\text{set} \\
10. \quad \text{return } G_0
\]

Now, all the introduced ingredients can be put together to elaborate the proposed method. Given an initial network structure \(G_0\) (only edges directed from the mode variable to other monitored variables exist.), historical data \(D\) and an untrained CNN value function approximator \(g\), the RLDGS learning algorithm iterates \(k\) times. In each iteration, the number of structures to be evaluated is obtained by equation (10) and (11), and then all possible new structures are obtained by applying operations (\(add\), \(delete\) and \(reverse\)) to \(G_0\) (invalid structures with cycles are ignored). After estimating their scores by \(g\), only the top \(N\) structures and \(G_0\) are evaluated by sampled data batch \(D_\text{batch}\). \(G_0\) is substituted by the best structure in the evaluation and the evaluated scores are fed back into \(\text{score}_\text{set}\) to optimize the value function approximator \(g\).
The optimization of $g$ can be implemented by gradient based methods [35]–[40]. Because score_set is not a big data set, to make $g$ converge fast enough, the whole score_set is used without sampling data batch. Besides, $g$ can also be optimized multiple times in each outer iteration. Finally, $G_0$ is returned as the learned structure. Algorithm 2 gives the pseudo code, where the parameter learning is not explicitly given because CPT parameters can be easily got by frequency [13].

Another thing that the pseudo code does not mention is that not all operations can be applied to obtain a valid structure. If there is no edge between two nodes, apparently reverse operation is invalid. And if an operation results in a cycle structure, the structure is ignored as well.

We will see that compared with the original greedy search method, the propose method distinctly costs less time after the value-function converges.

IV. CASE STUDY

Two systems, a 10-tank system and the classic Tennessee Eastman process (TEP) [41], are employed to demonstrate our method. We build Bayesian network diagnosers over $V = \{m\} \cup V_m$, where $m$ is the variable representing system health mode and $V_m$ is the set of monitored variables. The mode with the maximal a posterior probability, which can be obtained by (12), is the estimated mode.

$$P(m|v_0, v_1 \ldots v_{n-1}) \propto P(m) \prod_{i=0}^{n-1} P(v_i|Pa(v_i)). \quad (12)$$

We only employ BIC score because it is widely accepted in Bayesian network learning [13], [23] (Compared with the likelihood score, BIC score is able to effectively reduce overfitting), and this work focuses on improving the learning process by reinforcement learning.

A. THE 10-TANK SYSTEM

This subsection employs a 10-tank system, as illustrated in Figure 8, to demonstrate the proposed method. There are 10 tanks, 1 input pipe, 1 output pipe and 9 connection pipes in the system. The 9 connection pipes connect the three tanks at the bottom. The input pipe feeds water to the first tank. Water flows from tank $i$ to tank $i+1$ and finally outputs. Initially, all tanks are empty, and the input pipe is open. After some time, the height of the first tank increases to $h$ and the input pipe is closed. As water flow away, the height of the first tank decreases to $l$ and the input pipe is open again. The switch always occurs so that the height of the first tank keeps between $l$ and $h$.

Equations (13)–(16) describe the behaviors of the system where $A$ is the cross-sectional area of the tanks and $Sp$ is the cross-sectional area of the pipes. There are 20 faults considered in this system, 10 stuck faults and 10 leakage faults, where Stuck and Leakage represent fault the parameters respectively.

$$\frac{dh_i}{dt} = \frac{q_{i-1} - q_i - q_6}{A}, \quad i = 1, 2, \ldots, 10, \quad (13)$$

The 10-tank system is a typical system in process industry. Sometimes it is difficult to establish an accurate Bayesian network diagnoser because of some practical factors. Learning such a diagnoser is challenging due to the number of monitored variables. In addition to the input flow $q_0$, there are 21 monitored variables in the system. Accordingly, there are at most $2^{21} \approx 2.1 \times 10^6$ possible structures for BN diagnosers. The original greedy search based learning method evaluates $21 \times 20 + 1 = 421$ of them in each iteration. Although learning a good Bayesian network diagnoser is costly for the system, our method is capable of reducing much time.

The input flow, heights of the 10 tanks, and flows of the last 10 pipes with 20dB noise are used for diagnosis. All the monitored values were discretized into 10 discrete values by unsupervised k-means approach offered by Scikit-learn [42].

In our experiments, the value function is a CNN with four CNN layers and four max-pooling layers. All CNN layers padded the input data with zeros so that each output channel had the same size with the input channel. The first three CNN layers were used to extract structure features, which had 32, 16 and 8 channels respectively and their kernel sizes were 3. The last CNN layer had only one channel, and its kernel size was 1. All channels were merged by the last CNN layer. The kernel size of all pooling layers was 2. The average value of the outputs of the last pooling layer are the final output of the value-function.

The experiment was conducted on a personal computer with an Intel i7–4790 CPU and 12G memory. The greedy search was implemented by Python and the value-function was implemented by Pytorch [43]. In the search process, $N_0$ was set as the maximal number ($21 \times 20 = 420$) of possible new structures, just the same with the greedy search, and $N_1$ was 64. $k_0$ was set as the first iteration after the training loss of the value function was less than 3, and $k_1$ was set as $k_0 + 4$ and $k_0 + 8$. All the experiments were repeated 5 times to get robust results.

To demonstrate our method, a learned Bayesian network is given, and two factors are compared: learning time and
diagnosis accuracy. The former is used to show the efficiency improvement and the latter one gives how much diagnosis accuracy we lose because of the reduced evaluation number.

1) THE LEARNED BAYESIAN NETWORKS
Figure 9 gives the learned BN by RLDGS with \( k_1 - k_0 = 8 \) in one experiment. In the BN, most connected nodes had similar indexes. Actually, if two monitored variables were close to each other in the system, they had similar indexes. The figure indicates that the proposed approach is capable of finding the relationship between different variables and offering a good BN for the system. The original greedy search method and RLDGS with \( k_1 - k_0 = 4 \) also had similar results, but their BNs were not shown in this work to avoid redundancy.

2) THE TRAINING PROCESS
The scores and losses in our experiments are illustrated in Figure 10, where the horizontal axis represents the training time (minute), the left vertical axis is the scores evaluated by (1) (normalized by \( M \)) and the right vertical axis represents the losses of the value function. There are five curves in this figure: three of them are the scores of the original greedy search method, RLDGS with \( k_1 - k_0 = 8 \) and RLDGS with \( k_1 - k_0 = 4 \), and the other two curves gives the losses of the two RLDGS methods. The bars in the figure indicates the 95% confidence interval obtained from the 5 repetitions. In the first 20 min, the three methods had a similar increasing trend and the losses converged to a small number. After that, the score curves started to get apart and converged in different speeds. According to the three score curves, we can see that the greedy search method converged very slowly (\( \approx 65 \) min) and the RLDGS methods converged much faster (\( \approx 35 \) min for \( k_1 - k_0 = 4 \), and \( \approx 45 \) min for \( k_1 - k_0 = 8 \)). Although the greedy search method achieved the best score, the scores of the other two methods were still good enough.

The training time was illustrated more clearly by Figure 11 where the horizontal axis is the iteration times and the vertical axis represents the time. As the iteration increased, the time cost of the greedy search increased faster because the structures became more complex which were harder to be evaluated. However, the training time increased much slower because less structures were evaluated.

The specific number of evaluated structures are illustrated in Figure 12. As we can see, the number of evaluated likely structures decreased. That is because as the current
best structures became more complex, an operation (delete, add and reverse) is easier to cause a cycle. Finally, the GS algorithm steadily evaluated about 330 likely structures. The two RLDGS algorithms evaluated almost the same number with the GS, however, the number decreased quickly (with different slopes) when the value-function converged, and finally estimated 65 structures. After 15 iterations, about 265 structures were eliminated by the value-function.

The three figures indicate that the proposed method is able to learn a good BN with much less time.

3) THE DIAGNOSIS ACCURACY
To evaluate the diagnosis performance of the learned BNs, we give the receiver operating characteristics (ROC) curve of each BN and Figure 13 gives one ROC curve of the greedy search method. The area under the ROC curve (AUC) [44] is adopted as the quantitative criteria. AUCs of the three methods are compared in Figure 14 where the bars indicate the 95% confidence interval. The AUCs were very close to each other, which means the BNs learned by the three methods have very close diagnosis performance. The bars tell that the AUC of the greedy search method was more stable because the predicted score by the value function is not exactly the same with the real value.

4) THE INSTINCT OF THE CNN VALUE FUNCTION
To analyze how the CNN value function estimates scores, we visualize an input matrix and outputs of some kernels in a CNN value function by the heatmap in Figure 15. Dark blocks represent high outputs and light blocks are for low outputs. Subfigure a) is the input matrix, b) and c) are the outputs of two kernels in the first CNN layer, and d) is the output of a kernel in the second layer. The outputs of kernels in the first layer had the same shape with the input matrix. In subfigure b) and c), dark blocks located around the dark blocks in a) but with different patterns. This relationship means that different kernels in the first layer used distinct criteria to evaluate local structures. The output of the kernel in the second layer had a reduced shape, however, the distribution of dark blocks was similar to a). The similarity indicates that kernels in the second layer merged the information from the first layer and a block in d) represents a bigger local structure in a). In this way, the CNN value function merged and mapped local structure features into global structure features layer by layer and predicted the scores.
of collecting sufficient statistics from data and training the
using a value function approximator.
forcement learning fully makes use of all past experience
icy is first-order Markovian, where the “first-order” means
reinforcement learning into it. The classic greedy search pol-
ices is intractable because there are at most 2
probability distributions. However, building network struc-
Bayesian networks are a powerful tool to represent complex
ment learning directed greedy search is capable of rapidly
are monitored.
We made experiments on a laptop with Intel i9-8950K
CPU and 32G memory. Greedy search failed to build a
Bayesian network diagnoser because of overtime. So, we only
employed RLDGS with $k_1-k_0=4$. All continuous variables
were discretized into 8 discrete values.
The learned Bayesian network is not given because it is too
complex. The loss of the v-function and score of structures
are given in Figure 16 and the ROC curve (AUC=0.9021)
is shown in Figure 17. The characteristics of the figures are
similar to the ones in the former case.
The two experiments prove that the proposed reinforcement
learning directed greedy search is capable of rapidly
learning a good Bayesian network diagnoser.

V. ANALYSIS AND DISCUSSION
Bayesian networks are a powerful tool to represent complex
probability distributions. However, building network struc-
tures is intractable because there are at most $2^n$ possible
structures.

Traditional score-based structure learning algorithms tries
to solve this problem by searching, but the huge search space
is always a difficult obstacle. This work views the traditional
greedy search in a Markov decision process and integrates
reinforcement learning into it. The classic greedy search pol-
icy is first-order Markovian, where the “first-order” means
that the memory length is 1. No other information can be
obtained from the evaluations before the last time-step. Rein-
force ment learning fully makes use of all past experience
using a value function approximator.
The validity of the proposed method depends on the cost
of collecting sufficient statistics from data and training the
value function. If the training is much easier than collecting
statistics, our method theoretically outperforms the original
one. Fortunately, compared with the historical data, the evalu-
ated scores are much smaller. So, we do not need a complex
neural network. In addition that CNNs is easy to be trained
parallelly [34], training the value function approximator is not
difficult as demonstrated in the experiment.

Quantitatively, assume that there are $n$ monitored variables,
$a$ is the cost of evaluating a structure by data and $b$ is the
cost of training the value-function in each iteration. The
greedy search evaluates $O(n^2)$ structures in each iteration and
its complexity of the $k$ iterations is $O(akn^2)$. The proposed
method evaluates different number of structures in different
phases: $N_0$ for the first $k_0$ iterations, $N_0 \sim N_1$ for iteration
$k_0 \sim k_1$, and $N_1$ for the rest iterations. So, the whole
complexity is $O(a(k_0N_0+(k_1-k_0)(N_0+N_1)/2+(k-k_1)N_1)+kb)$. When $k \gg k_1, N_1 \ll n^2$, and $a \gg b$, the cost of our method
can be reduced as $(O(akN_1))$ which is significantly less than
the original greedy search method.

Compared with the original method, our method is
improved in two aspects.

- All information in the search process is fully utilized.
The traditional method only focuses on the current best
structure. Our method compiles all the information into
the value function.
- Training cost is significantly reduced by omitting impos-
sible structures. Based on the learned value function,
structures with a low estimated score (given by the value
function) are not evaluated by data, avoiding unneces-
sary computation.

VI. CONCLUSION
In this work, we propose an efficient Bayesian network
learning approach. This approach combines reinforcement
learning and greedy search, where a CNN is adopted as
the value-function. The initial learning process is almost
the same with the traditional greedy search method, but a
value-function is trained at the same time. When the training
loss of the value function is less than a threshold, the function
is employed to reduce the number of structures to be evalu-
ated. For the evaluation is more expensive than the training
of the value function, the proposed method is able to cost less
time and improve the efficiency.

A 10-tank system and the classic Tennessee East-
man process are employed to demonstrate the proposed
approach. In the first experiment, our method converged
much faster than the original one (reduced by 30%–50%)
with little score loss. In addition, the diagnosis accuracy also
shows that the Bayesian networks learned by the new method
were as good as the ones from the greedy search. In the second
case, the traditional greedy search failed because of over-
time, while the proposed one worked well. As a conclusion,
the proposed method is capable of learning accurate Bayesian
network diagnosers rapidly for complex systems.

In the future, we will use our approach to learn dynamic
Bayesian networks and try to combine reinforcement learning

FIGURE 17. The ROC curve of RLDGS.

B. THE TENNESSEE EASTMAN PROCESS
The Tennessee Eastman process is a typical industrial process
which is mainly composed of five process units: a two-phase
reactor, a separator, a stripper, a compressor and a mixer.
Martin-Villalba et al. [41] introduced how to implement it in
Modelica. There are 21 faults in the system and 52 variables
are monitored.

There are 21 faults in the system and 52 variables
are monitored.

The Tennessee Eastman process is a typical industrial process
improved in two aspects.

- All information in the search process is fully utilized.
The traditional method only focuses on the current best
structure. Our method compiles all the information into
the value function.
- Training cost is significantly reduced by omitting impos-
sible structures. Based on the learned value function,
structures with a low estimated score (given by the value
function) are not evaluated by data, avoiding unneces-
sary computation.

VI. CONCLUSION
In this work, we propose an efficient Bayesian network
learning approach. This approach combines reinforcement
learning and greedy search, where a CNN is adopted as
the value-function. The initial learning process is almost
the same with the traditional greedy search method, but a
value-function is trained at the same time. When the training
loss of the value function is less than a threshold, the function
is employed to reduce the number of structures to be evalu-
ated. For the evaluation is more expensive than the training
of the value function, the proposed method is able to cost less
time and improve the efficiency.

A 10-tank system and the classic Tennessee East-
man process are employed to demonstrate the proposed
approach. In the first experiment, our method converged
much faster than the original one (reduced by 30%–50%)
with little score loss. In addition, the diagnosis accuracy also
shows that the Bayesian networks learned by the new method
were as good as the ones from the greedy search. In the second
case, the traditional greedy search failed because of over-
time, while the proposed one worked well. As a conclusion,
the proposed method is capable of learning accurate Bayesian
network diagnosers rapidly for complex systems.

In the future, we will use our approach to learn dynamic
Bayesian networks and try to combine reinforcement learning

VOLUME 8, 2020

2821
with other basic structure learning methods, such as genetic methods, to achieve a better performance.

REFERENCES

[1] J. Sun, C. Li, C. Liu, Z. Gong, and R. Wang, “A data-driven health indicator extraction method for aircraft air conditioning system health monitoring,” Chin. J. Aeronaut., vol. 32, no. 2, pp. 409–416, 2018.

[2] C. Svärd, M. Nyberg, E. Frisk, and M. Krysander, “A data-driven and probabilistic approach to residual evaluation for fault diagnosis,” in Proc. IEEE Conf. Decis. Control, no. 1, Dec. 2011, pp. 95–102.

[3] Y. Hua, Q. Li, Z. Ren, and C. Liu, “A data driven method for quantitative fault diagnosability evaluation,” in Proc. 28th Chin. Control Decis. Conf. (CCDC), no. 6, May 2016, pp. 1890–1894.

[4] H. Khorasangi, “Model-and data-driven approaches to fault detection and isolation in complex systems,” Vanderbilt Univ., Nashville, TN, USA, Tech. Rep., 2018.

[5] S. X. Ding, Data-Driven Design of Fault Diagnosis and Fault-Tolerant Control Systems. 2014.

[6] H. Li and D.-Y. Xiao, “Survey on data driven fault diagnosis methods,” Kongzhi ya Juece/Control Decis., vol. 26, no. 1, pp. 1–9 and 16, 2011.

[7] B. Cui, Y. Liu, Q. Fan, Y. Zhang, Z. Liu, S. Yu, and R. Ji, “Multi-source information fusion based fault diagnosis of ground-source heat pump using Bayesian network,” Appl. Energy, vol. 114, pp. 1–9, Feb. 2014.

[8] D. L. C. Mack, G. Biswas, and X. D. Koutsoukos, “Learning Bayesian network structures to augment aircraft diagnostic reference models,” IEEE Trans. Automa. Sci. Eng., vol. 14, no. 1, pp. 358–369, Jan. 2017.

[9] D. Codetta-Raiteri and L. Portinale, “An introduction to deep reinforcement learning,” Mach. Learn., vol. 101, pp. 186–208, Sep. 2016.

[10] M. Hu, H. Chen, L. Shen, G. Li, Y. Guo, H. Li, J. Li, and W. Hu, “A machine learning Bayesian network for refrigerant charge faults of variable refrigerant flow air conditioning system,” Energy Build., vol. 114, pp. 668–676, Jun. 2018.

[11] D. Koller and N. Friedman, Probabilistic Graphical Models. 2009.

[12] C. M. Bishop, Pattern Recognition and Machine Learning, vol. 53, no. 9. Cambridge, MA, USA, 2013.

[13] M. He and D. He, “Deep learning based approach for bearing fault diagnosis using deep convolutional neural networks,” in Proc. ICLR, 2015, pp. 1–5.

[14] M. Milanič, W. Jia, J. S. Nelson, and B. Majoaran, Numerical optimization, vol. 43, no. 2, 2011.

[15] M. D. Zeiler, “ADADELFIA: An adaptive learning rate method,” 2012, arXiv:1212.5701.

[16] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015, arXiv:1502.03167.

[17] S. Ruder, “An overview of gradient descent optimization algorithms,” 2016, arXiv:1609.04747.

[18] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. ICLR, 2015, pp. 1–5.

[19] S. Boyd and L. Vandenberghe, Convex Optimization, vol. 25, no. 3, 2010.

[20] C. Martin-Villalba, A. Urquia, and G. Shao, “Implementations of the convex optimization,” IEEE Trans. Neural Netw., vol. 28, no. 3, pp. 523–533, Mar. 2017.

[21] A. P. Bradley, “The use of the area under the ROC curve in the evaluation of machine learning algorithms,” Pattern Recognit., vol. 30, no. 7, pp. 1145–1159, 1997.

[22] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 818–833.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commum. ACM, vol. 60, no. 6, pp. 84–90, Jun. 2017.

[24] M. Hu, H. Chen, L. Shen, G. Li, Y. Guo, H. Li, J. Li, and W. Hu, “A machine learning Bayesian network for refrigerant charge faults of variable refrigerant flow air conditioning system,” Energy Build., vol. 45, no. 1, pp. 13–24, Oct. 2017.

[25] M. Milanič, W. Jia, J. S. Nelson, and B. Majoaran, Numerical optimization, vol. 43, no. 2, 2011.

[26] M. D. Zeiler, “ADADELFIA: An adaptive learning rate method,” 2012, arXiv:1212.5701.

[27] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015, arXiv:1502.03167.

[28] S. Ruder, “An overview of gradient descent optimization algorithms,” 2016, arXiv:1609.04747.

[29] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. ICLR, 2015, pp. 1–5.

[30] S. Boyd and L. Vandenberghe, Convex Optimization, vol. 25, no. 3, 2010.

[31] C. Martin-Villalba, A. Urquia, and G. Shao, “Implementations of the convex optimization,” IEEE Trans. Neural Netw., vol. 28, no. 3, pp. 523–533, Mar. 2017.

[32] A. P. Bradley, “The use of the area under the ROC curve in the evaluation of machine learning algorithms,” Pattern Recognit., vol. 30, no. 7, pp. 1145–1159, 1997.
WENQUAN FENG received the Ph.D. degree in communication and information systems from Beihang University, Beijing, China. He is currently a Professor with the Department of Electronic and Information Engineering, Beihang University. His research interest includes automatic testing and fault diagnosis for complex systems and satellite navigation.

HONGBO ZHAO received the Ph.D. degree from Beihang University, Beijing, China, in 2012. He was an Academic Visitor with Nottingham University and then a Research Staff in Ohio University. He is currently an Associate Professor with the Department of Electronic and Information Engineering, Beihang University. His current research interests include satellite communication and data processing.

QI ZHAO received the Ph.D. degree in communication and information systems from Beihang University. She was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Pittsburgh, from 2014 to 2015. She is currently an Associate Professor of information and communication engineering with the School of Electronic and Information Engineering, Beihang University. She conducts research in intelligent systems in hybrid modeling, simulation, pattern recognition, image recognition and processing, and communication signal processing.