Bayesian Network Learning

Jiaqi Yang
Department of Computer Science
Graduate Center, CUNY
jyang2@gradcenter.cuny.edu

May 24, 2019

Abstract

Bayesian networks are a widely used graphical model with diverse applications in classification, prediction.To get the optimal Bayesian networks, lots of methods have been applied including ILP, Dynamic Programming, breadth-first BnB search, and A* searching. The running time and space used for learning a structure depend on the efficient of the algorithm applied, and also depend on how massive the dataset is. There are two main ways to get relation structure of an input dataset: local search and global search. As we can tell from the name, local search represents a method to find the local best Bayesian networks and the global search represents to find one of the optimal networks. Optimal networks are not limited to only one structure because if the input data is large enough, multiple best solutions with the same score can be optimal at the same time. Finding out one solution is good enough for the global search, while for local search running time would be improved but cannot guarantee the best outcome. At the same time, neural network learning similar combinatorial optimization problem has been proposed in many papers.

Index Terms —Bayesian Network, learning Bayesian structures, Score-based search, Neural Network, Combinatorial Optimization.

1 Introduction

Applying Bayesian networks to real-world problems typically requires building graphical representations of the problems. The probabilistic graphical model is applied for classification and prediction. One of the most popular method in last decades was dynamic programming, however it is usually out of time or space dealing with huge variables. Recently learning Bayesian networks is considered in several different ways, like formulating it as a shortest path searching optimization problem. Meanwhile, there are many similar combinatorial optimization problems, TSP for example, has been solved using neural network and reinforcement learning.

Table 1: Comparison of Algorithms.

Year	Investigator(s)	Area of focus
1992	Cooper and Herskovits [1]	An approximation method and introduction of learning Bayesian networks.
1996	Chickering *et al.* [2]	Prove learning Bayesian networks is NP-complete problem.
2003	Chichering *et al* [14]	Large-Sample Learning of Bayesian Networks is NP-Hard.
2004	Koivisto and Sood[20]	First time dynamic programming method was proposed to learn Bayes net structures.
2005	Moore *et al.* [3]	Dynamic programming method with scoring metrics of equation introduced to learn the networks.
2005	Teyssier and Koller [4]	Using pre-determined variable orderings to learn local maximum networks.
2000	Tian,J *et al.*[5]	an efficient depth-first branch-and-bound algorithm for learning Bayesian network structures.
2015	Beek and Hoffmann [6]	more constraints are applied to depth-first BnB approach for more efficient solving the Bayesian network learning problem.
2011	Malone, B *et al.* [7]	Improvement of memory-efficient and running time on Dynamic programming method.
2011	Changhe, Y *et al.* [8]	A* search for learning optimal network.
2013	Changhe, Y *et al.* [9]	Shortest path perspective with improved heuristic, which can be applied to other algorithms.
2014	Fan, X *et al.* [10]	Improve the A* search by detecting sparse structure before searching.
2017	Hanjun, D *et al.* [24]	Graph embedding network, structure2vec, to represent the actual steps.

The score-based search methods to find high-scoring structures for given data [Cooper and Herskovits, 1992; Heckerman, 1998] is widely used. Since learning a Bayesian network from data is NP-hard even if the number of parents per vertex in the DAG is limited to two, early approaches are mostly approximation methods [Cooper and Herskovits, 1992; Friedman et al., 1999; Heckerman, 1998]. Unfortunately, the models found by local search methods are not with a good quality guarantee.

Global search algorithms for learning a Bayesian network form data developed over the past several decades including dynamic programming[Koivisto and Sood, 2004; Silander and Myllymaki, 2006; Singh and Moore, 2005], integer linear programming, A* search, DFBnB search, and BFBnB search. For dynamic programming, the main idea is to solve small subproblems first and then apply the results to larger problems until a global learning problem is solved. However, these algorithms may be inefficient due to their need to fully evaluate an exponential solution space.

A recent constraint-based depth-first BnB search algorithm [Peter van Beek and Hella-Franziska Hoffmann] was proposed to find a valid Bayesian network. This algorithm was shown to be very efficient than dynamic programming, and regular depth-first BnB search. We believe its symmetry-breaking models can be modified and then applied to other more complex algorithms; such a formulation will make the size of the search space more efficient.

Numerous important problems can be framed as learning from graph data. Learning convolutional neural networks for arbitrary graphs[Mathias et al., 2016] is proposed. They present a general approach to extracting locally connected regions from graphs. They demonstrate that the learned feature representations are competitive and that their computation is highly efficient.

In this survey, I will present a collection of studies on learning Bayesian networks from different angles of view. At the same time, I will demonstrate the development of some methods. At the same time, I will demonstrate the neural network, Pointer Network, and Deep Q-learning. The approach has not been evaluated for Bayesian network structure learning. I will illustrate their core algorithm here without experiment results. The approaches for Bayesian network learning are evaluated on a representative suite of benchmark data. I will compare the advantages and disadvantages of all the algorithms in the end. In my survey, basic Bayesian theory and three themes are listed in TABLE 1.

2 BACKGROUND

There are not many surveys on learning Bayesian structure. Some of them collected papers which analyze a broad view of learning[17]. While my survey focuses on the particular task of learning Bayesian network structure from fully observed data using a search-and-score approach in both global and local views, including neural network related topics.

Score functions will be introduced in this section, and how the score-based searching for Bayesian networks defined as NP-hard problem will also be discussed here. After the background discussion, I will go to different type of algorithms in details.

2.1 Learning Networks and Score-based Functions

There are some popular score-base functions for learning Bayesian networks.The problem becomes finding the optimal structures with higher score for given data. All of theses functions are expressed as a sum over the individual variables, and the property is called decomposability. The most used functions include: Bayesian Dirichlet Family(BD), Minimum Description Length(MDL), Factorized Normalized Maximum Likelihood(fNML) etc.

The space of Bayesian networks with n nodes has $2^{O(n^2)}$ possible structures, and the problem of finding the best structure is in NP-hard.

Since the possible network structures to search are exponential in the number of variables even if an ordering on the variables is given, Cooper and Herskovits developed a greedy search algorithm called K2, and then Bouckaert[21] replaced the Bayesian scoring function in the K2 algorithm with a MDL scoring function.

MDL scoring function becomes the most widely used one recently. X_i represents each variables and PA_i is the parent sets of the variable. MDL computes the probability of each variable with all the rest variables combination sets. The score is computed by Entropy and penalty parameters(for smoothing).

$$\begin{aligned} MDL(X_i|PA_i) &= H(X_i|PA_i) + \frac{\log N}{2} K(X_i|PA_i) \\ H(X_i|PA_i) &= -\sum_{x_i, pa_i} N_{x_i,pa_i} \log \frac{N_{x_i,pa_i}}{N_{pa_i}} \\ K(X_i|PA_i) &= (r_i - 1) \prod_{X_i \in PA_i} r_l \\ MDL(G) &= \sum_i MDL(X_i|PA_i) \end{aligned}$$

The score is decomposable, so we do not have to recompute the score of the entire network when local changes happened which can affect only a small number of variables. The total score of the whole network consists of each score from variable given one of its parent sets.

2.2 Learning Networks and NP-Hard Problem

One of the earliest algorithms for learning Bayesian Network structure was done by F. Cooper and E. Herskovits [1] in 1992, and Heckerman [13] introduced the popular approach using score-based methods to find high-scoring structures for given data in 1998. The paper proposed by Cooper and Herskovits presents a Bayesian method for constructing probabilistic networks from databases. It is one of the earlies score-based searching conceptions for learning probability structures. It shows how to perform probabilistic inference by averaging over the inferences of multiple belief networks, which provide us with some local best belief networks.

Heckerman introduced a Bayesian metric(the BDe metric), that computes the relative posterior probability of a network structure given data. Based on that, learning Bayesian networks is NP-complete was proposed by Chickering *et al.* [2]. David Maxwell Chickering proved that the search problem of identifying a Bayesian network with a score metric is NP-complete problem, though limited at most K parents for each node.

Then in 2003, David[14] provided new complexity results for algorithms that learn discrete variable Bayesian networks from data. In the large-sample version of the learning problem, the learning Bayesian network problem reduced a known NP-complete problem. They showed that learning is NP-hard using a reduction from a restricted version of the NP-complete problem FEEDBACK ARC SET. The general FEEDBACK ARC SET problem is stated by Garey and Johnson (1979), and then they refer to a restricted version as DEGREEBOUNDED FEEDBACK ARC SET.

They all proved that it's NP-hard even the in degree is set to small number ($k > 1$), which means each variable can only select certain number of other variables as parents.

3 Ordering-based local search

Most approaches use local search(hill climbing) in the space of acyclic digraphs, which is time and space saving. Previous works focus on how to find out structures which are closer to the best solutions. Therefore, many Bayesian networks will be learned during the search process, and then choose the one with highest score among them.

Because of the BN-learning problem has been proved is NP-hard, so the search over the space of orderings instead of structures can be easy to implement. At the same time, it can give good results and the ordering can be arranged optimally in advance.

Local search over Bayesian Network structures was suggested by Chickering, Geiger, and Heckerman [15] in 1995. The algorithm improves a network iteratively by performing local modifications: adding an edge, removing an edge, or flipping an edge (reversing its direction). To alleviate the problem of local maxima, hill climbing is often augmented with random restarts or simulated annealing.

In 1999, Friedman et al. [16] proposed the Sparse-Candidate algorithm — a way to accelerate learning BN structures from data sets with many variables. The algorithm restricts the set of possible parents for each node by using several metrics. It then learns a BN subject to these restrictions, using a traditional hill-climbing algorithm. Finally, the algorithm uses the newly learned network to update the sets of candidate parents, and the entire procedure repeats. This algorithm performs faster than traditional (unrestricted) hill climbing on large data sets, and the quality of the learned network decreases only slightly.

Friedman and Koller proposed a new approach which efficiently compute a sum over the exponential number of networks that are consistent with a fixed order over networks variables in 2000. For a given order, both the marginal probability of the data and the posterior of a feature can be computed.

Marc Teyssier and Daphne Koller[4] presented another algorithm for learning BN-structures in 2012. Instead of searching over the space of structure, they proposed that search over the space of node orderings. Because highest-scoring BN consistent with a particular order and the order can be found in polynomial time, the simply method improved the searching speed largely. They used a greedy hill climber with tabu lists and random restarts to search over orders. At the same time, they pruned lots of possible parents for each node using some strategies to further reduce running time.

4 Dynamic Programming

Several exact algorithms based on dynamic programming were developed to learn optimal Bayesian networks[Koivisto and Sood, 2004; Singh and Moore, 2005; Silander and Myllymaki, 2006]. The main idea is to solve small subproblems until a global learning problem is solved. These algorithms cannot be always efficient due to their need to fully evaluate an exponential solution space.

The idea of using DP for exact Bayes net structure discovery was first proposed by Koivisto and Sood[20] in 2004. Their paper extends the work of Friedman and Koller [18][19], which computes the Bayesian posterior probability of structural features, as well as a single MAP model, by MCMC search over orderings.

In Singh and Moore, 2005[3] paper, they describe a algorithm for finding a Bayesian network that corresponds to a global maxima of a decomposable scoring function, such as BDeu or BIC. This algorithm is feasible for $n < 26$, and it is useful for learning moderately sized networks. From then, it becomes possible to study the properties of decomposable scoring functions without the potential for results being distorted by the behavior of local search.

Compared to Friedman and Koller's method, Singh and Moore have a difference class of functions to optimize, the decomposable scoring metrics. They have shown the incrementally removing leaves yields a different recursive equation, which can be solved by dynamic programming. Find the best set of parents from a set of potential parents is the goal for learning structures.

In 2011, Malone, Yuan,and Hansen[7] described a memory-efficient implementation of a dynamic programming algorithm. It leverages the layered structure of the dynamic programming graphs representing the recursive decomposition of the problem to reduce the memory requirements from $O(n2^n)$ to $O(C(n, n/2))$. The layered version of DP is more memory-efficient and faster than other approaches, such as the one presented by Singh and Moore(2005). Freeing the memory increases the size of learnable networks. They also applied theoretical properties of the MDL scoring function and calculate scores in a top-down manner to significantly improve the running time of the algorithm.

5 Shortest-path Concept Search

Learning Bayesian networks can be formulated as a shortest -path problem. The Depth-first BnB search[5] and its improvement with more constraints to pruning branches[6] consider the searching process as a depth-first search to get structure with lowest cost.

A* search algorithm[8][9][10] is introduced to solve the problem. It is similar with Breadth-first BnB search, so some pruning methods can also be used on both of them. Sparse-detecting, symmetry-breaking, and heuristic-tightening are gradually introduced to A* search. It is keeping improved and achieving better running time.

Suzuki[22] developed a branch-and-bound algorithm using MDL scoring function, which exhaustively searches through all network structures and guarantees to find the best scored network.

In 2000, Jin Tian[5] introduced an efficient depth-first branch-and-bound algorithm for learning Bayesian network structures, based on the MDL principle, for a given variable ordering. They extended Suzuki's work and present a more efficient method. A greedy search is applied before the BnB procedure to speed up the pruning process.

Yuan proposed learning optimal Bayesian networks using A* search in 2011. This paper formulates learning optimal Bayesian network as a shortest path finding problem. With the guidance of a consistent heuristic, the algorithm learns an optimal Bayesian network by only searching the most promising parts of the solution space. A* search algorithm significantly improves the time and space efficiency compared to dynamic programming algorithms.

In 2012, Brandon Malone and Yuan[23] developed an improved admissible heuristic that tries to avoid directed cycles within small groups of variables. Before this paper, A* and breadth-first branch and bound (BFBnB) were developed based on a simple admissible heuristic for learning Bayesian network structures that optimize a scoring function. However, the simple heuristic may contain many directed cycles and result in a loose bound. After they introduce a better admissible heuristic, the capability and speed for A* and BFBnB search improve significantly.

Beek and Hoffmann presented a constraint-based depth-first BnB approach for solving the Bayesian network learning problem in 2015. They proposed an improved constraint model that includes powerful dominance constraints, symmetry-breaking constraints, and cost-based pruning rules for effectively pruning the search for a minimum cost solution to the model. The constraints model they introduced also can be applied to other methods, which might help prune paths before or during searching process. At the same time, they also applied the heuristic provided by Brandon and Yuan(2012), which is used as lower bound to speed up solution finding.

6 Neural Network Related Combinatorial Problems

Vinyals, *et al.*(2015b) proposes training a pointer network using a supervised loss function comprising conditional log-likelihood, which factors into a cross entropy objective between the network's output probabilities and the targets provided by a TSP solver. The idea can be found in Figure 1.

The pointer network solves combinatorial optimization problems, and works on variable sized inputs something the baseline models (seq2seq with or without attention) cannot do directly. It turns out that the results outperform the baselines on fixed input size problems.

Irwan, *et al.*(2017) claimed that learning from examples in such a way is

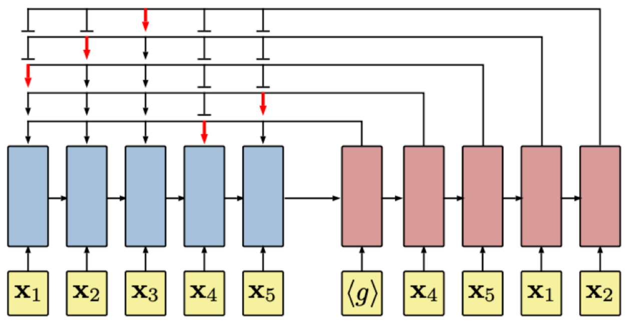

Figure 1: A pointer network architecture introduced by (Vinyals et al., 2015b).

undesirable for NP-hard problems because (1) the performance of the model is tied to the quality of the supervised labels, (2) getting high-quality labeled data is expensive and may be infeasible for new problem statements, (3) one cares more about finding a competitive solution more than replicating the results of another algorithm. Then they proposes learning combinatorial optimization with reinforcement learning.

Eugene B.*et al.*(2017) proposed structure discovery of undirected graphical models from observation data. They consider MRF edge estimation as a learnable function. The input data is standardized first and then sampled covariance matrix is estimated. In the next step, the matrix is embedded into multiple dilated convolutional networks, and then the edges are predicted in the end. All of those methods would be a inspiration for Bayesian network learning combined with neural network method.

7 Comparison of Different Algorithms

I collect papers into three different themes in this paper. In each section, local search, dynamic programming, and shortest-path view, papers are listed following the time line and discussed what the improvement is, comparing with the previous works.

In this section, I would like to illustrate the difference among these three directions for learning Bayesian networks in Table 2.

For scalability, the 'Great' in local search is relatively speaking compared with other two themes. In the shortest-path view theme, the scalability and running time are both depending on which algorithm used, because depth-first search belongs to it has no memory problem, but breadth-first and A* search do have the problem.

In the running time comparison, local search method usually runs the shortest time since it does not require optimal solution and the ordering search is much faster than the space search. DP has to iteratively solve all the small parts of

Comparison of Themes				
Theme Name	Scalability	Running Time	Variable Size	Solution
Local Search	Great	Fast	hundreds	non-optimal
DP	OM	Slow/OT	30	optimal
Shortest-path View	Depends	Depends	30-60	optimal

Table 2: OM is out of Memory; OT is out of time; Depends means depend on algorithm.

problem, so the running time is slower than local search, and sometimes out of time for large size of variables. The running time of shortest-path view is alway between the local search and DP.

Local search does not need to find out the best solution, therefore algorithms in this theme can deal with problems with hundreds of variables, sometimes more if with degree restriction. While dynamic programming and shortest-path searching algorithms can only deal with 30 to 60 variables, depending on which method is used. Depth-first search give the best network structure with more variables compared with Breadth-first optimal search. Large size of variable not only causes the problem of out of searching time, but also the out of memory results.

8 Future Directions

There are two mainly directions on my future research work, including both local maximum and global optimization searching. First, I will work on figuring out some properties of initial structures before learning. When some specific features are found via observing the dependency situation, the arcs among weak connected variables can be broken easily. If we can break the variables into different sparse groups, then the searching algorithms can be used on the groups at the same time. Therefore, the scalability problem would be solved dramatically.

Second direction for me is trying to use neural network method solving this problem. Some learning combinatorial optimization problems, like TSP, has been solved from the aspect of reinforcement learning with RNN[11]. At the same time, sparse geographic model Markov Random Field also was learned using Neural Network[12]. While this direction focus on the learning approximate connections.

8.1 Supervised Learning

In addition to the described pointer network, I would like to implement and train a pointer network with supervised learning, similarly to [Vinyals, *et al.*, 2015b]. The complete training process is illustrated in Figure 2. The loss is calculated by SHD(structural Hamming distance) between constructed structure from neural network and ground true structure. This loss is sent back to the neural network for further weights learning.

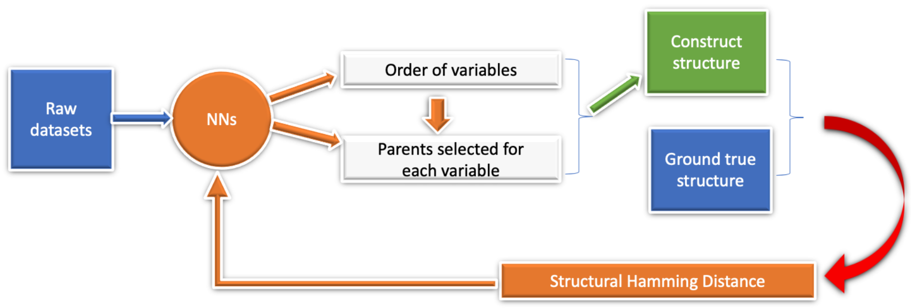

Figure 2: Supervised learning process structure.

For neural network learning direction, it is much harder to learn Bayesian networks than TSP optimal tours. Therefore, I figure out I might try to train the model using unsupervised learning(or call it self-supervised) to help find better structure with higher score.

8.2 Unsupervised Learning

Based on the good experiment results on TSP from Google Brain paper, Reinforcement Learning (RL) would be a good method providing an appropriate paradigm for training neural networks for combinatorial optimization. For Bayesian network learning, it has relatively simple reward mechanisms that could be even used at test time. Model-free Deep Q-learning Reinforcement Learning is a good way to optimize the parameters of a pointer network denoted θ.

The training objective is the expected structure MDL scores, which given an ordering can get the score from possible parents table in polynomial time. During training, stochastic gradient descent should be used to optimize the parameters. While because of the combinatorial possible status for training, then I will consider implement Actor-Critic algorithm to improve learning.

References

[1] F. Cooper and E. Herskovits(1992). A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 9:309–347.

[2] Chickering D. M.(1996): Learning Bayesian networks is NP-complete. In Learning from Data: Artificial Intelligence and Statistics, 121–130.

[3] Singh, A., and Moore, A.(2005): Finding optimal Bayesian Networks by Dynamic Programming. Technical report, Carnegie Mellon University.

[4] Marc Teyssier and Daphne Koller(2005): Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In: Proc. of UAI. 548–549

[5] Tian, J.(2000): A branch-and-bound algorithm for MDL learning Bayesian networks. In: Proc. of UAI. 580–588

[6] Peter van Beek and Hella-Franziska Hoffmann(2015): Machine learning of Bayesian networks using constraint programming.

[7] Malone, B., Yuan, C., Hansen, E.A.(2011): Memory-efficient dynamic programming for learning optimal Bayesian networks. In: Proc. of AAAI. 1057–1062

[8] Changhe Yuan, Brandon Malone and Xiaojian Wu(2011): Learning Optimal Bayesian Networks Using A* Search. 22nd International Joint Conference on Artificial Intelligence (IJCAI-11). Barcelona, Catalonia, Spain.

[9] Changhe Yuan, Brandon Malone(2013): Learning Optimal Bayesian Networks: A Shortest Path Perspective. Journal of Artificial Intelligence Research (JAIR).

[10] Fan,X.,Malone,B.,Yuan,C.(2014):Finding optimal Bayesian network structures with constraints learned from data. In: Proc. of UAI.

[11] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, Samy Bengio In(2017): Neural Combinatorial Optimization with Reinforcement Learning.

[12] Eugene B., Kyle K., Gaël V., Matthew B.(2017): Learning to Discover Sparse Graphical Models.

[13] David Heckerman(1998): A Tutorial on Learning with Bayesian Networks. Studies in Computational Intelligence, pages 33– 82. Springer Berlin / Heidelberg.

[14] Chickering D. M., Meek C., Heckerman D.(2003) :Large-Sample Learning of Bayesian Networks IS NP-Hard. UAI

[15] Chickering D. M. , Geiger D. , and Heckerman D.(1995): Learning Bayesian networks: Search methods and experimental results. Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics, pages 112–128.

[16] Nir Friedman, Iftach Nachman, and Dana Peer(1999): Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence(UAI-99), pages 206–215.

[17] Constantin Berzan(2012): An Exploration of Structure Learning in Bayesian Networks.

[18] Friedman, N., Koller, D.(2000): Being Bayesian about Network Structure. UAI-16.

[19] Friedman, N., Koller, D.(2003): Being Bayesian about Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks. Machine Learning, 50, 95–125.

[20] Koivisto, M., Sood, K. (2004). Exact Bayesian Structure Discovery in Bayesian Networks.

[21] R. R. Bouckaert(1994): Probabilistic Network Construction Using the Minimum Description Length Principle.

[22] J. Suzuki(1996): Learning Bayesian Belief Networks Based on the Minimum Description Length Principle: An efficient algorithm using the BnB technique. In L. Saitta, editor, Proceedings of the Thirteenth International Conference on Machine Learning, pages 462-470.

[23] Changhe Yuan, Brandon Malone(2012): An Improved Admissible Heuristic for Learning Optimal Bayesian Networks.In Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-12).

[24] Hanjun Dai, Elias B. Khalil, Yuyu Zhang, Bistra Dilkina, Le SongLearning(2017): Combinatorial Optimization Algorithms over Graphs.

[25] Mathias Niepert, Mohamed Ahmed, Konstantin Kutzkov (2016): Learning Convolutional Neural Networks for Graphs.

[26] Oriol Vinyals, Meire Fortunato, Navdeep Jaitly (2016b): Pointer Networks.

[27] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, Samy Bengio (2017):Neural Combinatorial Optimization with Reinforcement Learning.