

一种增加先验知识库的贝叶斯网络推理模型

瞿锡垚¹, 刘学军¹, 张礼²

(1. 南京航空航天大学 计算机科学与技术学院, 江苏 南京 211106;

2. 南京林业大学 信息科学技术学院, 江苏 南京 210037)

摘要: 贝叶斯网络作为一种不确定知识表示网络, 由网络结构和各节点的条件概率表组成, 在解决系统决策问题方面具有先天的理论优势。目前在大多数贝叶斯网络的应用中, 各节点条件概率表的产生均是以人工输入的方式完成, 这在一些拥有较多网络节点的复杂背景中, 需要巨大的人工消耗, 效率低下。针对这一问题, 提出一种增加先验知识库的贝叶斯网络推理模型。根据具体的建模问题创建先验知识库, 在该先验知识库下对网络节点进行类别标记, 然后根据局部马尔可夫性自动生成各节点的条件概率表。在贝叶斯网络推理任务中, 使用在精确推理任务中处理速度快、应用最为广泛的联结树算法, 并使用 Hugin 算法完成消息的传递。最后通过一个贝叶斯网络实例验证了整个模型的处理流程。

关键词: 贝叶斯网络; 知识库; 局部马尔可夫性质; 联结树算法; Hugin 消息传递

中图分类号: TP31

文献标识码: A

文章编号: 1673-629X(2019)08-0092-04

doi:10.3969/j.issn.1673-629X.2019.08.018

An Inference Model for Bayesian Network with Prior Knowledge Base

QU Xi-yao¹, LIU Xue-jun¹, ZHANG Li²

(1. School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China;

2. School of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China)

Abstract: As an uncertain knowledge representation network, Bayesian network is composed of network structure and conditional probability table of each node and has an innate theoretical advantage in solving decision problems. At present, in most applications of Bayesian network, the generation of conditional probability table of each node is completed in the form of manual input, which requires huge labor consumption and low efficiency in some complex backgrounds with many network nodes. To address this problem, an inference model for Bayesian network with prior knowledge base is proposed. A prior knowledge base is created for the specific modeling problem, based on which the network nodes are labeled with classes, and then the conditional probability table for each node are generated automatically according to local Markov properties. To infer Bayesian network, the clique tree algorithm with efficiency and popularity in precise inference is adopted, and the Hugin algorithm is utilized to make message transmission. Finally, the entire processing flow of the model is verified by a Bayesian network example.

Key words: Bayesian network; knowledge base; local Markov property; clique tree algorithm; Hugin message passing

0 引言

贝叶斯网络^[1-3]是一种基于贝叶斯定理的概率图模型, 可表示为 $\text{BN} = \langle \langle V, E \rangle, P \rangle$ 。网络结构由 $\langle V, E \rangle$ 表示, 是一个有向无环图, 其中 V 是网络节点, E 是节点之间有向边, P 指代条件概率表 (conditional probability table, CPT)。贝叶斯网络同时具有概率论和图论的理论基础, 拥有不确定知识的表

示能力。概率推理 (probabilistic inference) 和最大后验概率解释 (maximum a posteriori explanation) 是贝叶斯网络推理的两个基本问题^[4]。概率推理过程分为精确推理^[5-7]和近似推理^[8-9]两种, 分别适用于网络复杂度低和复杂度高的场景。最大后验概率解释则是对建模问题的计算结果进行解释性说明与分析。贝叶斯网络在被提出之初, 由于其需要大量的计算且被证明是 NP

收稿日期: 2018-09-20

修回日期: 2019-01-22

网络出版时间: 2019-03-27

基金项目: 国家自然科学基金 (61802193); 江苏省自然科学基金 (BK20170934)

作者简介: 瞿锡垚 (1994-), 男 (苗), 硕士研究生, CCF 会员 (89038G), 研究方向为机器学习及应用; 刘学军, 博士, 教授, 博导, 研究方向为机器学习及应用。

网络出版地址: <http://kns.cnki.net/kcms/detail/61.1450.TP.20190327.1620.014.html>

难问题^[10],很长一段时间无法得到广泛应用。直到计算机的出现,让贝叶斯网络重新得到重视,并广泛应用于故障诊断^[11-12]、神经网络^[13]以及作战重心评估^[14]等问题。

目前对于贝叶斯网络的应用中,大多是在给定网络结构和条件概率表之后,根据证据节点计算其他节点的概率值。而互联网时代带来了数据量的剧增,导致基于贝叶斯网络的系统决策问题复杂度提升,即贝叶斯网络结构中节点数和有效边数增加。在条件概率表的构造过程中,增加一个节点会带来两种新的状态,即该节点发生或不发生,而将这两种新状态加入到原始网络结构中时,构造条件概率表的工作量将面临指数级的增加,这在一定程度上限制了贝叶斯网络的应用范围。而在实际应用中,贝叶斯网络节点之间并非都是完全不一致的,一些节点之间具有较高的相似性。

针对网络节点增加带来的问题以及节点之间相似的应用现实,文中提出并实现了一种增加先验知识库的贝叶斯网络推理模型。根据建模问题相关专家给出的节点类别,以及类别之间的条件概率,建立先验知识库,自动生成贝叶斯网络的条件概率表,使用联结树精确推理算法^[15]完成贝叶斯网络的概率推理。

1 先验知识库

贝叶斯网络是由一个有向无环图(directed acyclic graph, DAG)和图中各节点的条件概率表构成,两者兼备时,才能进行网络推理。其中条件概率表记录的是节点 V_i 与其对应的父节点 V^{pa} 的关联关系。当节点 V_i 入度为零时(即没有父节点),该节点的条件概率表存放其设定的先验概率;当节点 V_i 拥有多个父节点 V_j^{pa} 时,该节点条件概率表存放所有父节点对应的所有状态下的 V_i 各概率值。例如图1中拥有三个父节点的节点 D ,对于 $D = T$ 和 $D = F$ 时,均对应应有8个状态下的8个概率值。为减少大数据带来的巨大工作量,文中提出并实现了先验知识库,自动生成各节点的条件概率表。首先对图中所有节点添加类别标识,并对各类别添加先验概率以及各类别之间的条件概率,具体方式依据建模问题的背景知识。之后根据条件概率公式自动推理计算出各节点的条件概率表。具体推理过程如下:

图1展示了一个实例图,对于多入度节点 D ,条件概率公式如下:

$$\Pr(D | A, B, C) = \frac{\Pr(A, B, C, D)}{\Pr(A, B, C)} \quad (1)$$

对 $\Pr(A, B, C, D)$ 使用乘法公式展开有:

$$\Pr(A, B, C, D) = \Pr(A | B, C, D) * \Pr(B | C, D) * \Pr(C | D) * \Pr(D) \quad (2)$$

其中, $\Pr(C | D)$ 和 $\Pr(D)$ 均可以根据其类型信息从知识库中直接读取概率值,对于 $\Pr(A | B, C, D)$ 和 $\Pr(B | C, D)$, 根据局部马尔可夫性质(local Markov property), 条件概率只依赖于自己的直接父节点和直接子节点,因此:

$$\Pr(A | B, C, D) = \Pr(A | B, D) \quad (3)$$

$$\Pr(B | C, D) = \Pr(B | D) \quad (4)$$

式4已化简为可以直接从知识库中读取概率值的结果,对于式3,可以由条件概率 $\Pr(A | B)$ 拆分得到:

$$\Pr(A | B) = \Pr(A | B, D = F) + \Pr(A | B, D = T) \quad (5)$$

根据式2~5, $\Pr(A, B, C, D)$ 可化为:

$$\Pr(A, B, C, D) = \Pr(A | B, D) * \Pr(B | D) * \Pr(C | D) * \Pr(D) \quad (6)$$

对式1中分母采用相同的化简方式,可得:

$$\Pr(A, B, C) = \Pr(A | B) * \Pr(B | C) * \Pr(C) \quad (7)$$

最终式1化为:

$$\Pr(D | A, B, C) = \frac{\Pr(A | B, D) * \Pr(B | D) * \Pr(C | D) * \Pr(D)}{\Pr(A | B) * \Pr(B | C) * \Pr(C)} \quad (8)$$

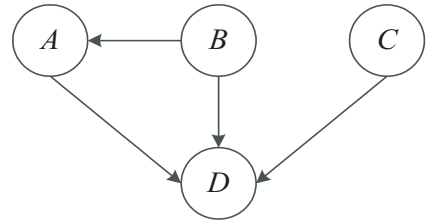


图1 条件概率表产生实例

根据上述推理方式,可以自动计算贝叶斯网络中所有节点的条件概率表,减少数据量增加带来的巨大工作量,让使用者能够将任务时间主要集中于贝叶斯网络的最大后验概率解释。

2 贝叶斯网络推理

确定贝叶斯网络结构和条件概率表之后,文中使用联结树算法对网络进行精确推理。联结树算法是目前贝叶斯网络精确推理任务中应用最广泛的算法,可以推理任何网络(单连通或者多连通),整个算法流程如图2所示。

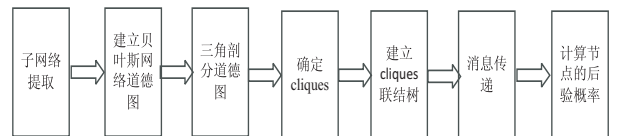


图2 联结树算法推理过程

子网络的提取过程是为了减少运算量,在给定证据之后,可以删除与证据无关的节点,即证据的祖先节

点中不包含的点。在贝叶斯网络中依次用无向边连接每两个具有共同子节点的节点,然后去除所有贝叶斯网络中有向边的方向便得到贝叶斯网络的道德图。之后使用 Kjaerulff 算法对道德图进行三角剖分,该算法具体过程如下:

算法 1:Kjaerulff 算法。

输入:贝叶斯网络道德图;

输出:原道德图中所有完全子图。

步骤 1:复制原贝叶斯网络 BN 到 BN^* ,在 BN^* 中选择具有最小权重的节点 V ,权重定义为与该点直接相连的边数,若有多个节点权重相同,则选择需要增加边数最小的节点。

步骤 2:在 BN^* 中依次连接与节点 V 直接相连的节点,同时在 BN 中做同样的连接,得到一个含有 V 的完全子图。

步骤 3:在 BN^* 中删除节点 V 以及与其相连接的边。

步骤 4:重复以上步骤,直到 BN^* 中没有节点为止。

经过三角剖分得到的每一个完全子图即为一个 clique。每一个 clique 均可看作是由贝叶斯网络中一组节点组成的一棵树,两个 clique 中关于贝叶斯网络节点的交集为一个分割集。分割集是连接 clique 的关键,将所有的分割集插入 clique 之间可组建一个联结树。标记已确定的 clique 为 C_i , $S_{C_i C_j}$ 表示 C_i 与 C_j 的分割集, S 表示分割集集合,其算法具体过程如下:

算法 2:建立 clique 联结树算法。

输入:clique 集合与分割集集合;

输出:clique 联结树。

步骤 1:在 S 中选择一个具有最大质量的分割集 $S_{C_i C_j}$ (质量的定义是指分割集中元素的个数)。

步骤 2:若 C_i 和 C_j 属于两棵不同的树,则将 $S_{C_i C_j}$ 插入 C_i 和 C_j 中,若属于同一棵树则重新选择分割集, $S_{C_i C_j}$ 的插入把两棵树合并为一棵树。

步骤 3:重复以上步骤直到所有分割集均被插入。

建立好联结树之后是消息的传递,文中使用计算速度快的 Hugin 消息传递算法^[16]。消息传递是为了让联结树达到全局一致,在该状态下,一个 clique 的势函数(用 φ 表示)即为该 clique 中所有节点与证据节点 V_e 的联合概率。对于贝叶斯网络中任一节点 V ,找到包含该节点的 clique,则该节点在给定证据下的后验概率为:

$$P(V | V_e) = \frac{P(V, V_e)}{\sum_V P(V, V_e)} \quad (9)$$

Hugin 算法的具体过程如下:

算法 3:Hugin 算法。

输入:clique 联结树,证据节点 V_e 以及条件概率表;

输出:贝叶斯网络中所有节点对于证据节点的后验概率。

步骤 1:初始化算法参数。对每个 clique 和分割集的势函数用 1 进行初始化,对任意一个节点 V ,选择一个包含 F_V (由节点 V 和它父亲节点构成的节点集)的 clique 节点 C ,更新 C 的势函数 $\varphi_C = \varphi_C \times P(V | V_{pa})$ 。其中 V_{pa} 表示节点 V 的父亲节点。

步骤 2:设置证据。选择一个包含证据的 clique 节点 C ,对其势函数进行更新: $\varphi_C = \varphi_C \times P(V = \text{False})$ 。

步骤 3:消息全局传递。这一步主要包含相邻 clique 节点之间的单一信息传递,证据收集 COLLECT_EVIDENCE(C) 和证据扩散 DISTRIBUTE_EVIDENCE(C) 三个过程。式 10 是单一信息传中的投影过程,式 11 是吸收过程。

$$\varphi_{C_i C_j} = \sum_{C_i - C_i \cap C_j} \varphi_{C_i} \quad (10)$$

$$\varphi_{C_j} = \varphi_{C_j} \times \frac{\varphi_{C_i C_j}^{\text{old}}}{\varphi_{C_i C_j}} \quad (11)$$

COLLECT_EVIDENCE(C) 过程首先选择一个 C 做标记,对 C 没做标记的邻近 clique 递归调用过程 COLLECT_EVIDENCE,传递信息从 C 到调用过程 COLLECT_EVIDENCE 的 clique。之后是 DISTRIBUTE_EVIDENCE(C) 过程,同样是选择一个 C 做标记,传递信息从 C 到每一个没做标记的邻近 clique,对 C 没做标记的邻近 clique 递归调用过程 DISTRIBUTE_EVIDENCE。

步骤 4:经过以上步骤会得到全局一致的联结树,使用式 9 即可计算贝叶斯网络节点在证据节点下的后验概率。

3 实例验证

这部分将使用一个简单的例子来描述增加先验知识库的贝叶斯推理模型的整个处理流程。假设知识库中有三个类分别为 a, b, c ,图 3 展示了三个类型各自的先验概率以及它们之间的条件概率。

没有起始节点的箭头表示指向节点的先验概率,例如 a 类节点的先验概率为 $\Pr(a = F) = 0.5$ 。有起始节点的箭头拥有两个概率值,分别代表起始节点两种不同状态,例如 a 类节点指向 b 类节点的箭头,表示在条件 b 下, a 发生的概率,概率值为 $\Pr(a = F | b = F) = 0.6$, $\Pr(a = F | b = T) = 0.5$ 。

在图 3 所代表的先验知识库下,文中建立一个简单的贝叶斯网络,如图 4 所示。

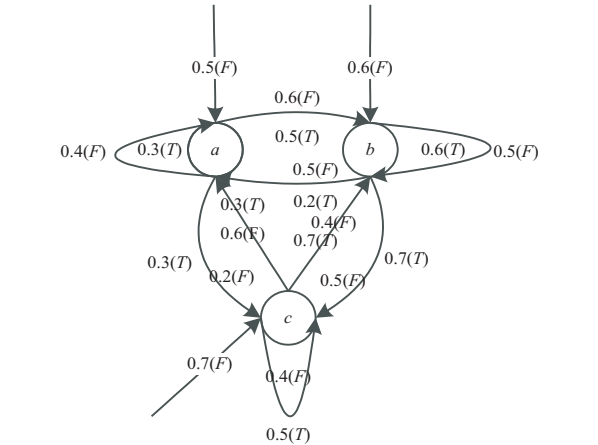


图 3 三个类的条件概率表

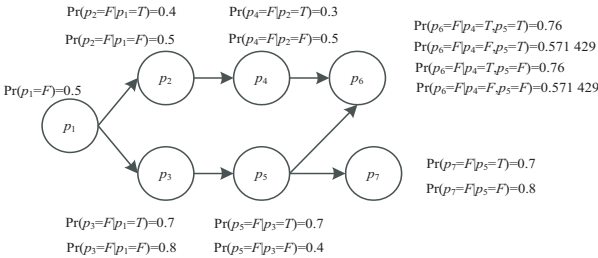


图 4 实例贝叶斯网络结构和条件概率表

a 类节点有 p_1, p_5 , b 类节点有 p_2, p_6 , c 类节点有 p_3, p_4, p_7 。设置节点 p_6 为证据节点,根据第 1 节中的推理方法可自动计算出各节点的条件概率表,图 4 中展示了其具体值。表 1 展示了贝叶斯网络的最终推理结果。

表 1 各节点在给定证据下的后验概率

节点	后验概率	节点	后验概率
p_1	0.50	p_5	0.50
p_2	0.55	p_6	1.00
p_3	0.25	p_7	0.00
p_4	0.73		

对于证据节点 p_6 ,节点 p_7 不在其祖先节点中,所以表 1 中节点 p_7 的后验概率为 0,节点 p_6 作为证据,后验概率为 1。表 1 的计算结果中节点 p_4 的后验概率最大,节点 p_3 的后验概率最小,说明在节点 p_6 发生的情况下,节点 p_4 最可能发生,而节点 p_3 最不可能发生。

4 结束语

文中提出了一种增加先验知识库的贝叶斯网络推理模型。对网络节点进行类别标记,并添加各类别的先验概率及各类别间的条件概率,在此基础上,利用局部马尔可夫性完成对各网络节点条件概率表的自动推理计算,大大减少了人工输入条件概率表的时间消耗。对于贝叶斯网络的推理,使用在精确推理任务中速度快、应用最为广泛的联结树算法,并使用了 Hugin 消息

传递,最终通过实例验证了整个模型的处理流程。

参考文献:

[1] 刘伟娜,霍利民,张立国. 贝叶斯网络精确推理算法的研究[J]. 微计算机信息,2006,22(9):92-94.

[2] 王华伟,周经伦,何祖玉,等. 基于贝叶斯网络的复杂系统故障诊断[J]. 计算机集成制造系统,2004,10(2):230-234.

[3] PEARL J. Fusion, propagation, and structuring in belief networks[J]. Artificial Intelligence, 1986, 29(3):241-288.

[4] 厉海涛,金光,周经伦,等. 贝叶斯网络推理算法综述[J]. 系统工程与电子技术,2008,30(5):935-939.

[5] DÍEZ F J. Local conditioning in Bayesian networks[J]. Artificial Intelligence, 1996, 87(1-2):1-20.

[6] LAURITZEN S L, SPIEGELHALTER D J. Local computations with probabilities on graphical structures and their application to expert systems[J]. Journal of the Royal Statistical Society, 1988, 50(2):157-224.

[7] SHACHTER R D, D'AMBROSIO B, DEL FAVERO B A. Symbolic probabilistic inference in belief networks[C]//Eighth national conference on artificial intelligence. Boston, Massachusetts: AAAI Press, 1990:126-131.

[8] DAGUM P, KARP R, LUBY M, et al. An optimal algorithm for Monte Carlo estimation[C]//Proceedings of IEEE 36th annual foundations of computer science. Milwaukee, WI, USA: IEEE, 1995:142-149.

[9] HENRION M. Search-based methods to bound diagnostic probabilities in very large belief nets[C]//Proceedings of the seventh conference (1991) on uncertainty in artificial intelligence. Los Angeles, California, USA: Morgan Kaufmann Publishers Inc., 1991:142-150.

[10] COOPER G F. The computational complexity of probabilistic inference using Bayesian belief networks (research note)[J]. Artificial Intelligence, 1990, 42(2-3):393-405.

[11] 王永强,律方成,李和明. 基于粗糙集理论和贝叶斯网络的电力变压器故障诊断方法[J]. 中国电机工程学报, 2006, 26(8):137-141.

[12] 姚成玉,陈东宁,王 斌. 基于 T-S 故障树和贝叶斯网络的模糊可靠性评估方法[J]. 机械工程学报, 2014, 50(2):193-201.

[13] ZAIDI N A, PETITJEAN F, WEBB G I. Preconditioning an artificial neural network using Naive Bayes[C]//Pacific-Aasia conference on knowledge discovery and data mining. [s. l.]: Springer International Publishing, 2016:341-353.

[14] 李正浩,刘学军. 一种基于贝叶斯网络的作战重心评估模型[J]. 计算机技术与发展, 2014, 24(9):50-53.

[15] 刘俊娜. 贝叶斯网络推理算法研究[D]. 合肥:合肥工业大学, 2007.

[16] JENSEN F V, LAURITZEN S L, OLESEN K G. Bayesian updating in causal probabilistic networks by local computations[J]. Computational Statistics Quarterly, 1990, 4:269-282.