

基于自编码器的贝叶斯网嵌入及概率推理*

杜斯^{1,2}, 祁志卫^{1,2}, 岳昆^{1,2}, 段亮^{1,2}, 王笏辉^{1,2}

¹(云南大学信息学院, 云南昆明 650500)

²(云南省智能系统与计算重点实验室(云南大学), 云南昆明 650500)

通信作者: 岳昆, E-mail: kyue@ynu.edu.cn



摘要: 贝叶斯网 (BN) 是不确定性知识表示和推理的基本框架, 广泛用于社交网络、知识图谱和医疗诊断等领域. 特定领域中基于 BN 的分析诊断和决策支持, 其核心计算任务是基于 BN 进行多次概率推理. 然而, 使用传统的概率推理方法, 基于同一 BN 的多次概率推理其中间过程存在很多重复的计算结果, 具有较高的时间复杂度. 为了提高多次概率推理的效率, 提出易于重用和易于计算的贝叶斯网嵌入及相应的概率推理方法. 首先, 借鉴图嵌入的基本思想, 使用点互信息矩阵来表示 BN 的有向无环图结构和条件概率参数, 提出基于自编码器和注意力机制的 BN 嵌入方法. 其中, 自编码器的每一编码层利用节点与其邻居节点 (父节点和子节点) 的相关性生成节点嵌入, 从而在嵌入向量中保存 BN 节点间的概率依赖关系. 然后, 使用嵌入向量之间的距离来度量节点之间的联合概率, 提出基于嵌入向量的 BN 概率推理方法. 实验证明, 针对 BN 的多次概率推理, 所提方法的效率高于现有方法, 且能得到准确的推理结果.

关键词: 贝叶斯网; 高效概率推理; 图嵌入; 自编码器; 注意力机制

中图法分类号: TP18

中文引用格式: 杜斯, 祁志卫, 岳昆, 段亮, 王笏辉. 基于自编码器的贝叶斯网嵌入及概率推理. 软件学报, 2023, 34(10): 4804–4820. <http://www.jos.org.cn/1000-9825/6670.htm>

英文引用格式: Du S, Qi ZW, Yue K, Duan L, Wang JH. Autoencoder-based Bayesian Network Embedding and Probabilistic Inferences. Ruan Jian Xue Bao/Journal of Software, 2023, 34(10): 4804–4820 (in Chinese). <http://www.jos.org.cn/1000-9825/6670.htm>

Autoencoder-based Bayesian Network Embedding and Probabilistic Inferences

DU Si^{1,2}, QI Zhi-Wei^{1,2}, YUE Kun^{1,2}, DUAN Liang^{1,2}, WANG Jia-Hui^{1,2}

¹(School of Information Science and Engineering, Yunnan University, Kunming 650500, China)

²(Yunnan Key Laboratory of Intelligent Systems and Computing (Yunnan University), Kunming 650500, China)

Abstract: Bayesian network (BN), as a preliminary framework for representing and inferring uncertain knowledge, is widely used in social network, knowledge graph, medical diagnosis, etc. The centric computing task of BN-based analysis, diagnosis, and decision-support in specific fields includes multiple probabilistic inferences. However, the high time complexity is doomed on the same BN by using the traditional inference methods, due to the several intermediate results of probability calculations that cannot be shared and reused among different inferences. Therefore, to improve the overall efficiency of multiple inferences on the same BN, this study proposes the method of BN embedding and corresponding probabilistic inferences. First, by incorporating the idea of graph embedding, the study proposes a BN embedding method based on the autoencoder and attention mechanism by transforming BN into the point mutual information matrix to preserve the directed a cyclic graph and conditional probability parameters simultaneously. Specifically, each coding layer of the autoencoder generates node embedding by using the correlation between a node and its neighbors (parent and child nodes) to preserve the

* 基金项目: 国家自然科学基金 (62002311); 云南省基础研究计划杰出青年项目 (2019FJ011); 云南省重大科技专项 (202002AD080002); 云南省基础研究项目 (202001BB050052)

收稿时间: 2021-10-01; 修改时间: 2021-12-22; 采用时间: 2022-02-26; jos 在线出版时间: 2023-04-04

CNKI 网络首发时间: 2023-04-06

probabilistic dependencies. Then, the method for probabilistic inferences to measure the joint probability by using the distance between embedding vectors is proposed. Experimental results show that the proposed method outperforms other state-of-the-art methods in efficiency, achieving accurate results of probabilistic inferences.

Key words: Bayesian network (BN); efficient probabilistic inference; graph embedding; autoencoder; attention mechanism

贝叶斯网 (Bayesian network, BN) 作为一种典型的用于不确定性知识表示和推理的概率图模型, 广泛用于医疗诊断^[1]、社交网络^[2]和知识图谱^[3,4]等领域. BN 由有向无环图 (directed acyclic graph, DAG) 和条件概率表 (conditional probability table, CPT) 组成, 每个节点表示一个随机变量, 每一条有向边代表两个节点之间的概率依赖关系, 其依赖程度由 CPT 中的概率参数值来描述. 基于 BN 的分析诊断和决策支持应用, 以 BN 的概率推理为基本计算任务, 即针对给定证据计算查询变量取值的条件概率. 例如, 基于图 1 中用于肺病诊断的 BN, 针对患者存在“呼吸困难 (dyspnea, D)”症状来推测其患“肺癌 (lung cancer, L)”的概率, 这一任务可转换为针对证据 $D=T$ (即 dyspnea 值为真) 计算查询变量 lung cancer 为真 (T) 的概率, 即 $P(L=T|D=T)$. 从实际应用的角度, 图 1 所示 BN 反映了一段时期内肺病诊断中相关变量之间存在的依赖关系, 基于该 BN 的诊断分析和决策支持, 实质上是基于该 BN 进行多次概率推理. 考虑以下两个肺病诊断任务, 在患者“X 射线”检查结果存在异常 (即 $X=T$) 的条件下患者“抽烟”的可能性, 以及 $X=T$ 且患者存在“呼吸困难”症状 (即 $D=T$) 的条件下“抽烟”的可能性, 可将其转换为计算 $P(S=T|X=T)$ 和 $P(S=T|X=T, D=T)$ 的概率推理任务, 其计算过程涉及的中间结果为 $P(S=T, X=T, D=T)$, 不难看出, 两次概率推理任务重复计算 $P(S=T, X=T, D=T)$, 而 $P(S=T, X=T, D=T)$ 的计算又涉及联合概率 $P(S=T, X=T, D=T, L, B)$ 的计算. 因此多次概率推理中 BN 变量个数越多且变量的可能取值 (即变量的势) 越多时, 重复计算的中间结果也就越多, 对概率推理效率的影响越显著. 因此, 如何避免多次概率推理中的重复计算、高效地实现诊断分析, 是基于 BN 的智能系统真正走向实用的重要保证, 也是本文拟研究解决的问题.

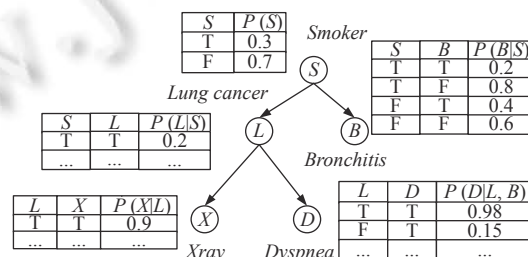


图 1 肺病诊断贝叶斯网

BN 概率推理主要分为精确推理和近似推理两类, 以变量消元法 (variable elimination) 为代表的精确推理算法和以吉布斯采样 (Gibbs sampling) 为代表的近似推理算法. 这两类 BN 概率推理方法的时间复杂度都随 BN 中节点数呈指数增长, 且均被证明为 NP 难问题^[5,6]. 如前所述, 基于同一 BN 的多次概率推理过程中存在大量重复的计算结果. 这意味着概率推理次数越多, 时间复杂度越高; 重复计算的中间结果越多, 效率就越低. 因此, 为了高效地实现基于同一 BN 的多次概率推理, 本文旨在研究中间结果易于重用和关键步骤易于计算的 BN 概率推理方法.

图嵌入 (graph embedding, GE)^[7,8]作为经典的图表示学习方法, 广泛应用于图节点分类、聚类和链接预测等图分析任务中. GE 旨在保存图结构和属性的条件下, 将图中的节点或边表示为低维向量^[9-13], 其高效性和易于计算性在许多图分析任务中都得到了验证^[13]. 针对 BN 多次概率推理效率低、中间结果重复计算等问题, 本文借鉴 GE 的思想, 将 BN 表示为一组低维嵌入向量, 通过使用嵌入向量保存节点间的概率依赖关系, 将概率推理转换为向量之间的计算, 以此提高 BN 多次概率推理的效率.

BN 中 DAG 和 CPT 分别从定性和定量的角度反映了节点间的概率依赖关系, 子节点依赖于父节点的概率由条件概率参数表示, 不同的节点取值对应 CPT 中不同的条件概率参数; 此外, 相邻节点中父节点也可能依赖于其子节点, 不直接相邻的两个节点之间也可能存在概率依赖关系. 然而, 现有的 GE 模型不能完整地表示 BN 中的概率依赖关系, 而目前研究人员提出的基于矩阵分解的 BN 嵌入及其概率推理方法^[14], 其嵌入过程中仍未考虑不相

邻节点之间的概率依赖关系,且多次概率推理效率受限于概率推理次数.

为了有效表示 BN 中节点不同取值对应的不同条件概率参数值,本文基于 BN 构建带权有向图,使用图中的节点表示 BN 中节点的不同取值状态,基于点互信息 (pointwise mutual information, PMI)^[15]将相邻节点之间的概率依赖关系转化为边上的权重.为了高效地构建带权有向图,且尽可能地保存 DAG 和 CPT 的信息,本文以各节点条件概率作为节点取值的采样概率,提出基于最大似然估计 (maximum likelihood estimation) 的 BN 采样方法,并近似计算 PMI 值,从而得到图中各边上的权重.

基于自编码器的 GE 方法,利用多个堆叠的非线性层来获取节点之间的复杂关系^[13].针对 BN 中相邻节点间的双向概率依赖关系,基于上述带权有向图中的带权邻接矩阵 (PMI 矩阵),本文借鉴基于自编码器的 GE 思想来生成 BN 节点的嵌入.为了进一步保存 BN 中不相邻节点之间的概率依赖关系,本文提出结合注意力机制^[16]的 BN 嵌入方法.其中,每个编码层都利用节点与其邻居节点的注意力系数,根据邻居节点来生成新的节点表示.由于 PMI 矩阵有效描述了 BN 中节点间的相关性,本文直接使用 PMI 矩阵中的值作为相邻节点的注意力系数.为了保存 BN 中概率依赖关系的方向性,分别使用父节点的向量表示和子节点的向量表示来生成新的节点向量表示.每个编码层均传递节点及其相邻节点的概率信息,经过堆叠的编码层,在各节点的向量表示中均保存了该节点与其他不同节点之间的概率依赖关系.

进一步,基于 BN 的嵌入向量,本文提出使用欧式距离来衡量节点之间相似性的方法,从而得到节点之间的联合概率,然后根据贝叶斯公式计算得到条件概率. BN 嵌入实现了使用一组低维向量来表示高维的概率图模型的方法,使得基于 BN 的概率计算更易于实现, BN 节点的嵌入向量可重复用于多次概率推理计算,通过向量间距离的计算,可高效地实现基于 BN 的概率推理.

本文第 1 节介绍 BN 概率推理及图嵌入相关工作.第 2 节给出自编码器和注意力机制的 BN 嵌入方法.第 3 节给出基于嵌入向量的 BN 概率推理方法.第 4 节给出实验结果及性能分析.第 5 节总结全文并展望未来的工作.

1 相关工作

1.1 BN 概率推理

传统的 BN 概率推理,主要包括精确推理和近似推理两类算法.变量消元法和团树传播 (clique-tree propagation) 是经典的精确推理算法,变量消元法通过不断分解联合概率来消去非证据变量和非查询变量,从而得到推理结果,而团树传播通过概率的局部传播实现概率推理,这两种算法均需要花费指数级计算时间^[5],因此不适用于实际中规模较大 BN 之上的多次概率推理.对此,人们提出以吉布斯采样和前向采样 (forward sampling) 为代表的近似推理算法,其根据 BN 中的先验分布来生成样本,通过对合格样本计数来得到近似概率推理结果^[6].近似推理算法在一定程度上解决了大规模 BN 的推理问题,但是这两种算法的有效性与样本的数量成正比、效率与样本数量成反比,多次概率推理涉及 CPT 上的重复查询和计算,且不同概率推理的中间结果不能重用.为了提高多次概率推理的效率,文献^[14]提出基于矩阵分解的 BN 嵌入及其概率推理方法,但其效率优势受限于概率推理的次数.

1.2 图嵌入

GE 作为经典的图表示学习方法,旨在保留图结构或属性的条件下将图中的每个节点或边映射为低维向量,以便用于后续图分析任务.基于矩阵分解的 GE 方法根据节点对之间的相似性来保持图的性质.基于随机游走的方法主要通过随机游走获得训练样本,利用 Skip-Gram 模型将每个节点映射到低维空间,早期主要以基于 DeepWalk^[17]和 Node2vec^[18]方法为代表;在此基础上, dynnode2vec^[19]通过 Node2vec 初始化快照,对动态变化的节点执行随机游走,从而更新动态变化信息;tNodeEmbed^[20]以类似句子嵌入的方式实现节点嵌入,获取节点角色和边的动态变化.为了获取高度非线性图结构,研究人员提出基于深度学习的 GE 方法,例如,SDNE^[13]利用深度堆叠自编码器来保持网络的一阶和二阶相似性,通过非线性函数获得最终嵌入向量;DNNGR^[21]使用堆叠去噪自编码器来获得图结构信息和嵌入向量;GALA^[22]采用完全对称的图卷积自编码器模型生成图的低维嵌入表示,编码器和解码器分别使用拉普拉斯平滑和拉普拉斯锐化,使得生成的嵌入向量能够同时保存图的结构信息和节点特征;DySAT^[23]通过

邻域结构和时间两个维度的联合自注意力来计算节点嵌入; GraphAIR^[24]通过领域聚合模块通过组合邻域特征来构建节点特征表示, 各邻域交互模块通过乘法运算来显示建模邻域信息的交互, 使得生成的节点嵌入向量能保留属性信息. 而目前考虑节点属性信息的 GE 方法中涉及的节点属性大多是文本属性、所属类别和节点标签等. 因此直接使用基于自编码器的 GE 方法不能保存 BN 的 CPT.

2 基于自编码器的贝叶斯网嵌入

2.1 PMI 矩阵的构建

给定一个 BN $B = (G, \theta)$, 其中 $G = (V, E)$ 表示 BN 的 DAG 结构, θ 表示 BN 中参数的集合, V 中每一个节点均有 $X_i = \{X_{ik^i}\}_{k^i=1}^{m_i}$, m_i 表示节点 X_i 的势, k^i 表示节点 X_i 的取值. BN 中节点不同取值对应不同的条件概率参数, 每个条件概率参数表示在其父节点的当前取值下该节点当前取值的概率. 为了同时保留 BN 中节点取值及其条件概率参数, 本文将 BN 转换为带权有向图, 其中节点表示 BN 中取不同值的节点, 边的权重由 BN 的 CPT 转换而来. 由于 PMI 值衡量了两个随机变量之间的相关性, 因此将 BN 中节点对的 PMI 值作为边的权重. 具体而言, 用 PMI 值 $\mathbf{W}(X_{ik^i}, X_{jk^j}) = \text{PMI}(X_{ik^i}, X_{jk^j})$ 来表示有向边 (X_{ik^i}, X_{jk^j}) 的权重, 考虑到带权有向图的方向性, 有 $\mathbf{W}(X_{ik^i}, X_{jk^j}) \neq \mathbf{W}(X_{jk^j}, X_{ik^i})$, 节点 X_{ik^i} 到节点 X_{jk^j} 的有向边的权重 $\mathbf{W}(X_{ik^i}, X_{jk^j})$ 的计算方法如下:

$$\mathbf{W}(X_{ik^i}, X_{jk^j}) = \text{PMI}(X_{ik^i}, X_{jk^j}) = \log \frac{P(X_{ik^i}, X_{jk^j})}{P(X_{ik^i})P(X_{jk^j})} \quad (1)$$

其中, $P(X_{ik^i}, X_{jk^j})$ 表示节点对 (X_{ik^i}, X_{jk^j}) 中两节点同时出现的概率, 用 $\#(X_{ik^i}, X_{jk^j})$ 来表示, 因此 $\mathbf{W}(X_{ik^i}, X_{jk^j})$ 的计算公式可转换为:

$$\mathbf{W}(X_{ik^i}, X_{jk^j}) = \text{PMI}(X_{ik^i}, X_{jk^j}) = \log \frac{P(X_{ik^i}, X_{jk^j})}{P(X_{ik^i})P(X_{jk^j})} = \log \frac{\#(X_{ik^i}, X_{jk^j})}{\#(X_{ik^i})\#(X_{jk^j})} \quad (2)$$

为了将 BN 中节点间的 CPT 转换为 PMI 值, 本文将 BN 中节点 CPT 中的条件概率参数作为节点采样概率, 提出基于最大似然估计的 BN 采样方法. 首先, 使用 BN 节点的拓扑排序作为节点采样顺序, 针对待采样的节点 X_i , 假设节点的势为 2, 记 $p = P(X_i = 0)$ 或 $p = P(X_i = 0 | \pi(X_i) = j)$, $\pi(X_i) = j$ 表示节点 X_i 的父节点集合 $\pi(X_i)$ 的取值组合数为 j , 根据 p 值生成随机数 $r \in [0, 1]$, 若 $r \in [0, p]$, 则采样节点的取值为 $k^i = 0$, 若 $r \in (p, 1]$ 则采样节点的取值为 $k^i = 1$, 当节点 X_i 的势大于 2 时以此类推. 为了保证采样样本能尽可能地包含 BN 中的全部信息, 将基于最大似然估计得到的条件概率参数与原 CPT 中的各参数进行比较, 若两者较为接近, 则说明采样样本足以描述原 BN 中的信息, 从而确定采样算法的生成样本数 n_s . 最大似然估计计算公式如下:

$$\theta_{ijk^i} = \frac{\#(X_{ijk^i})}{\#(X_{ij})} \quad (3)$$

其中, $\#(X_{ij})$ 表示满足 $\pi(X_i) = j$ 的样本数, $\#(X_{ijk^i})$ 表示满足 $X_i = k^i$ 和 $\pi(X_i) = j$ 的样本数.

基于采样得到的样本可计算 $\#(X_{ik^i}, X_{jk^j})$ 、 $\#(X_{ik^i})$ 和 $\#(X_{jk^j})$, 其中, $\#(X_{ik^i}, X_{jk^j})$ 表示同时满足 $X_i = k^i$ 和 $X_j = k^j$ 的样本数, $\#(X_{ik^i})$ 和 $\#(X_{jk^j})$ 分别表示满足 $X_i = k^i$ 和 $X_j = k^j$ 的样本数. 带权有向图中有向边和权重保存了 BN 的 DAG 和 CPT, 因此使用带权邻接矩阵 \mathbf{W} (PMI 矩阵) 来表示图的初始向量. 值得注意的是, 在构建 PMI 矩阵过程中, 将不存在有向边相连的节点对的 PMI 值设置为 0, 而存在有向边的节点对 PMI 值根据公式 (2) 来计算.

2.2 基于自编码器的 BN 节点嵌入

基于构建的带权有向图及生成的 PMI 矩阵, 本节引入注意力机制, 给出基于自编码器的 BN 嵌入方法 (auto-encoder based Bayesian network embedding, ANE), 如图 2 所示, 包括自编码器输入、编码和解码 3 个阶段.

(1) 输入

PMI 矩阵中的行向量表示每个节点作为父节点时与其子节点间的相关信息, 列向量表示每个节点作为子节点时与其父节点间的相关信息. 为了同时保留节点作为父节点和节点作为子节点与其他节点的相关性, 将 PMI 矩阵及其转置拼接在一起作为自编码器的输入. 具体针对节点 X_{ik^i} 来说, 输入自编码器的向量为 $\mathbf{x}_{ik^i}^{(0)} = [\mathbf{W}_{ik^i}, \mathbf{W}_{ik^i}^T]$.

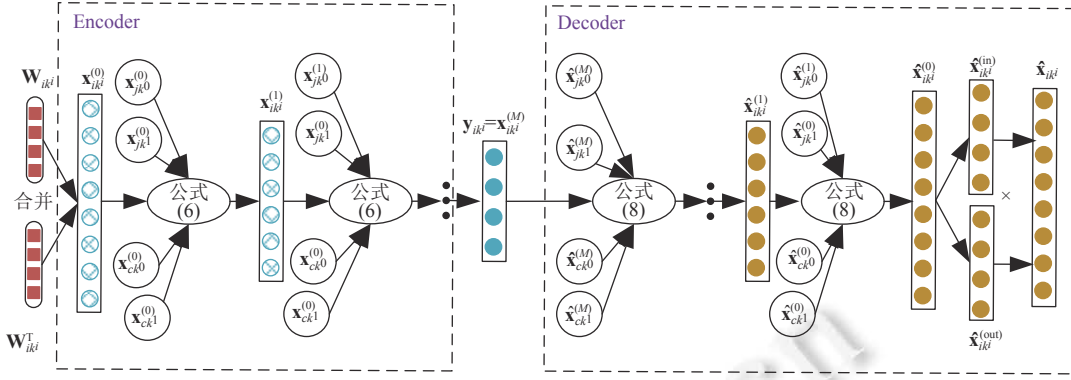


图2 基于自编码器的BN嵌入模型

(2) 编码

为了在编码阶段保存相邻节点间的双向概率依赖关系和不相邻节点间的概率依赖关系, 本文扩展了传统的注意力机制, 直接使用 PMI 矩阵中的值作为注意力机制中节点与其相邻节点的注意力系数. 由于 BN 中的相邻节点包括父节点和子节点, 且节点之间存在方向性, 因此分别利用子节点和父节点的向量表示来共同生成该节点的向量表示. BN 中节点 X_{ik^i} 与其子节点 X_{jk^i} 的相关表示为 $e_{ij}^{k^i k^i} = \mathbf{W}(X_{ik^i}, X_{jk^i})$, 为了使该节点在其多个子节点中的相关系数可比, 使用 *Softmax* 函数对节点与所有子节点的相关系数做正则化处理:

$$\alpha_{ij}^{k^i k^i} = \text{Softmax}_j(e_{ij}^{k^i k^i}) = \frac{\exp(e_{ij}^{k^i k^i})}{\sum_{l \in N_{i(\text{out})}} \exp(e_{il}^{k^i k^i})} \quad (4)$$

其中, $N_{i(\text{out})}$ 表示以节点 X_{ik^i} 为父节点时所包含的子节点及节点 X_{ik^i} 本身构成的集合.

类似地, $e_{ci}^{k^i k^i} = \mathbf{W}^T(X_{ik^i}, X_{ck^i})$ 表示节点 X_{ik^i} 与其父节点 X_{ck^i} 的相关性, 对其父节点的相关系数做正则化处理:

$$\alpha_{ci}^{k^i k^i} = \text{Softmax}_c(e_{ci}^{k^i k^i}) = \frac{\exp(e_{ci}^{k^i k^i})}{\sum_{l \in N_{i(\text{in})}} \exp(e_{li}^{k^i k^i})} \quad (5)$$

其中, $N_{i(\text{in})}$ 表示以节点 X_{ik^i} 为子节点时所包含的父节点及节点 X_{ik^i} 本身构成的集合.

节点 X_{ik^i} 输入自编码器的初始向量为 $\mathbf{x}_{ik^i}^{(0)}$, 第 m 层编码层通过整合父节点和子节点信息, 得到节点 X_{ik^i} 的第 m 层输出:

$$\mathbf{x}_{ik^i}^{(m)} = \sum_{j \in N_{i(\text{out})}} \alpha_{ij}^{(m)} \sigma(\mathbf{W}_p^{(m)} \mathbf{x}_{ik^i}^{(m-1)}) + \sum_{c \in N_{i(\text{in})}} \alpha_{ci}^{(m)} \sigma(\mathbf{W}_c^{(m)} \mathbf{x}_{ik^i}^{(m-1)}) \quad (6)$$

其中, $m \in \{1, 2, \dots, M\}$, $\mathbf{W}_p^{(m)}$ 和 $\mathbf{W}_c^{(m)}$ 分别为第 m 层编码层的参数, $\sigma(\cdot)$ 表示激活函数. 经过 M 层编码层的编码, 得到节点 X_{ik^i} 的嵌入向量表示, 即 $\mathbf{y}_{ik^i} = \mathbf{x}_{ik^i}^{(M)}$.

PMI 矩阵的维度为 BN 中各节点势的总和, 因此自编码器输入维度为 PMI 矩阵维度的两倍, 表示为 $2g$ ($g = \sum_{i=1}^n m_{X_i}$), 得到节点嵌入矩阵维度为 d ($d \ll g$). 带权有向图中 PMI 值越大的相邻节点, 表示节点间的相关性越大, 使用 O_{ik^i} 表示节点 X_{ik^i} 作为父节点与其子节点边的权重之和, 即 $O_{ik^i} = \sum_{l=1}^n \mathbf{W}(X_{ik^i}, X_{lk^i})$, I_{jk^i} 表示节点 X_{jk^i} 作为子节点时与其所有父节点边的权重之和, 即 $I_{jk^i} = \sum_{l=1}^n \mathbf{W}(X_{lk^i}, X_{jk^i})$. 相邻节点 X_{ik^i} 和 X_{jk^i} 中, 若父节点 X_{ik^i} 与子节点边的权重越大且子节点 X_{jk^i} 与其父节点边的权重越大, 则说明这两节点间存在概率依赖关系程度越强, 两个节点也就越相似. 因此本文使用 $\sqrt{(O_{ik^i} + 1)(I_{jk^i} + 1)}$ 来近似表示父节点 X_{ik^i} 和子节点 X_{jk^i} 之间的相似指标, 使用嵌入向量内积近似表示嵌入向量相似性, 为了保证相邻节点嵌入向量的相似性, 使用 *Sigmoid* 函数对两相似性数值进行规范化, 通过对数函数来最小化损失, 定义如下:

$$L_{1st} = \sum_{i=1}^n \sum_{j \in N_{i(\text{out})}} \log \frac{\text{Sigmoid}(\mathbf{y}_{ik^i}^T \mathbf{y}_{jk^i})}{\text{Sigmoid}(\sqrt{(O_{ik^i} + 1)(I_{jk^i} + 1)})} \quad (7)$$

其中, \mathbf{y}_{ik^i} 和 \mathbf{y}_{jk^j} 分别表示节点 X_{ik^i} 和节点 X_{jk^j} 的嵌入向量.

(3) 解码

为了在解码阶段生成有效的节点嵌入, 使用层数与编码器相同的解码器重构嵌入向量, 编码器的输出即为解码器的输入, 即 $\hat{\mathbf{x}}_{ik^i}^{(M)} = \mathbf{x}_{ik^i}^{(M)}$, 每个解码层都尝试重构相对应的编码层, 即每个解码层分别利用子节点和父节点的表示, 根据节点与其父节点和子节点间的相关性来重构节点表示. 针对节点 X_{ik^i} , 第 $m-1$ 层解码器重构了第 m 层的节点表示, 其输出为:

$$\hat{\mathbf{x}}_{ik^i}^{(m-1)} = \sum_{j \in N_{i(\text{out})}} \alpha_{ij}^{k^i k^j} \sigma(\hat{\mathbf{W}}_p^{(m-1)} \hat{\mathbf{x}}_{ik^i}^{(m)}) + \sum_{c \in N_{i(\text{in})}} \alpha_{ci}^{k^c k^i} \sigma(\hat{\mathbf{W}}_p^{(m-1)} \hat{\mathbf{x}}_{ik^i}^{(m)}) \quad (8)$$

其中, $m \in \{1, 2, \dots, M\}$, $\hat{\mathbf{W}}_p^{(m-1)}$ 和 $\hat{\mathbf{W}}_c^{(m-1)}$ 为第 $m-1$ 层解码层的参数.

经过 M 层解码器后得到输出 $\hat{\mathbf{x}}_{ik^i}^{(0)}$. 为了加快自编码器的训练速度、更好地提取 BN 方向性, 本文改进了解码器的解码网络结构, 将解码向量 $\hat{\mathbf{x}}_{ik^i}^{(0)}$ 分别经过两个规模完全相同的全连接网络, 得到节点作为子节点的解码向量表示 $\hat{\mathbf{x}}_{ik^i}^{(\text{in})}$ 和节点作为父节点的解码向量表示 $\hat{\mathbf{x}}_{ik^i}^{(\text{out})}$, 使用向量积 $\hat{\mathbf{x}}_{ik^i} = \hat{\mathbf{x}}_{ik^i}^{(\text{in})} \times \hat{\mathbf{x}}_{ik^i}^{(\text{out})}$ 作为最终的解码向量. 自编码器的目标是 minimize 编码输入和解码输出的重构损失, 损失函数如下:

$$L_{2\text{nd}} = \sum_{i=1}^n \|\hat{\mathbf{x}}_{ik^i} - \mathbf{x}_{ik^i}\|_2^2 \quad (9)$$

PMI 矩阵具有稀疏性, 输入矩阵 \mathbf{X} 中的非零元素数量远远小于零元素的数量, 导致在训练过程中更容易重构 \mathbf{X} 中零元素, 因此, 对非零元素的重构误差施加比零元素更大的惩罚, 修改后的目标函数如下:

$$L_{2\text{nd}} = \sum_{i=1}^n \|(\hat{\mathbf{x}}_{ik^i} - \mathbf{x}_{ik^i}) \odot b_i\|_2^2 = \|(\hat{\mathbf{X}} - \mathbf{X}) \odot B\|_2^2 \quad (10)$$

其中, \odot 表示哈达马积, $b_i = \{b_{i,j}\}_{j=1}^n$, 若 $\mathbf{X}[i, j] = 0$, 则 $b_{i,j} = 0$; 若 $\mathbf{X}[i, j] > 0$, 则 $b_{i,j} = \xi > 1$.

为了在嵌入向量中保存 BN 中相邻节点相似性, 同时最小化自编码器中的重构损失, 结合公式 (7) 和公式 (10), 得到的最终的损失函数为:

$$L = L_{2\text{nd}} + \lambda L_{1\text{st}} \quad (11)$$

其中, λ 为控制自定义损失函数的参数, 取值为 1.

2.3 基于自编码器的 BN 嵌入

基于自编码器的 BN 嵌入方法主要包括采样样本、生成 PMI 矩阵和基于自编码器生成嵌入向量 3 个部分, 具体实现见算法 1.

算法 1. 基于自编码器的 BN 嵌入.

输入: $B = (G, \theta)$, BN; P , 训练迭代次数; M , 编码器层数; n_s , 生成样本数; D , 基于最大似然估计的 BN 采样得到的样本集合;

输出: \mathbf{Y} , 嵌入向量矩阵; $\hat{\mathbf{X}}$, 重构矩阵.

1. 初始化参数 $\mathbf{W}_p = \{\mathbf{W}_p^{(1)}, \dots, \mathbf{W}_p^{(M)}, \hat{\mathbf{W}}_p^{(M)}, \dots, \hat{\mathbf{W}}_p^{(1)}\}$ 、 $\mathbf{W}_c = \{\mathbf{W}_c^{(1)}, \dots, \mathbf{W}_c^{(M)}, \hat{\mathbf{W}}_c^{(M)}, \dots, \hat{\mathbf{W}}_c^{(1)}\}$ 和 \mathbf{W}_f
2. 根据 D 和公式 (2) 计算 PMI 矩阵中的值, 构建 PMI 矩阵 \mathbf{W}
3. 输入矩阵: $\mathbf{X}^{(0)} \leftarrow [\mathbf{W}, \mathbf{W}^T]$
4. FOR $epoch=1$ TO P DO
5. FOR $m=1$ TO M DO
6. 根据公式 (4) 计算每个节点的 α_{ij} , 构成矩阵 $\mathbf{C}^{(p)}$
7. 根据公式 (5) 计算每个节点的 α_{ci} , 构成矩阵 $\mathbf{C}^{(c)}$
8. $\mathbf{X}^{(m)} \leftarrow \sigma(\mathbf{W}_p^{(m)} \mathbf{X}^{(m-1)}) \mathbf{C}^{(p)} + \sigma(\mathbf{W}_c^{(m)} \mathbf{X}^{(m-1)}) \mathbf{C}^{(c)}$
9. END FOR
10. $\hat{\mathbf{X}}^{(M)} \leftarrow \mathbf{X}^{(M)}$ //解码器的第 M 层输入

-
11. **FOR** $m = M$ **DOWNTO** 1 **DO**
 12. $\hat{\mathbf{X}}^{(m-1)} \leftarrow \sigma(\hat{\mathbf{W}}_p^{(m)} \hat{\mathbf{X}}^{(m)}) \mathbf{C}^{(p)} + \sigma(\hat{\mathbf{W}}_c^{(m)} \hat{\mathbf{X}}^{(m)}) \mathbf{C}^{(c)}$
 13. **END FOR**
 14. $\hat{\mathbf{X}}^{(0)}$ 通过自定义解码器结构结合该层参数 \mathbf{W}_f 分别得到 $\hat{\mathbf{X}}^{(\text{in})}$ 和 $\hat{\mathbf{X}}^{(\text{out})}$
 15. $\hat{\mathbf{X}} \leftarrow \hat{\mathbf{X}}^{(\text{in})} \times \hat{\mathbf{X}}^{(\text{out})}$
 16. 根据公式 (11), 使用 $\frac{\partial L}{\partial \mathbf{W}_p}$ 、 $\frac{\partial L}{\partial \mathbf{W}_c}$ 和 $\frac{\partial L}{\partial \mathbf{W}_f}$ 进行反向传播, 更新编码器和解码器的参数
 17. **END FOR**
 18. $\mathbf{Y} \leftarrow \hat{\mathbf{X}}^{(M)}$
 19. **RETURN** $\mathbf{Y}, \hat{\mathbf{X}}$
-

算法 1 的第 1 行生成样本的时间复杂度为 $O(n_s \times n)$, 空间复杂度为 $O(n_s \times n)$; 第 2 行中 PMI 矩阵的构造时间复杂度为 $O(n_s \times \log n_s)$, 若 BN 中每个节点的势的总和为 g ($g = \sum_{i=1}^n m_{X_i}$), 则 PMI 矩阵的大小为 $g \times g$, PMI 矩阵构造的空间复杂度就为 $O(g^2)$; 基于自编码器的 BN 嵌入, 首先对 PMI 矩阵的相关系数进行归一化, 时间复杂度为 $O(n^2)$, 自编码层生成节点向量的时间复杂度为 $O(n^2 \times d)$, 空间复杂度为 $O(n^2)$. 由于采样样本的数量 n_s 远大于 n 、 g 和 d , 因此, 算法 1 的时间复杂度为 $O(n_s \times n)$, 空间复杂度为 $O(n_s \times n)$.

3 基于嵌入向量的概率推理

通过 BN 嵌入, 使得节点间的概率依赖关系被保存在嵌入向量中, 这种概率依赖关系越大, 相应的 PMI 值就越大, 两个节点的嵌入向量就越接近, 意味着相关嵌入向量的空间距离就越短. 而空间距离越短, 说明两个节点所对应变量同时出现的概率越大, 则这两个节点的联合概率可近似表示为它们同时出现的概率, 因此, 我们使用嵌入向量之间的归一化欧式距离来近似表示节点对的联合概率.

首先, 节点对 (X_{ik^i}, X_{jk^j}) 嵌入向量之间的欧式距离定义如下:

$$D(\mathbf{y}_{ik^i}, \mathbf{y}_{jk^j}) = \sqrt{\sum_{l=1}^d (\mathbf{y}_{ik^i}^l - \mathbf{y}_{jk^j}^l)^2} \quad (12)$$

进而将以上欧式距离归一化到 $[0, 1]$, 有效表示 BN 节点对之间的联合概率:

$$P(X_{ik^i}, X_{jk^j}) \approx \text{sim}(\mathbf{y}_{ik^i}, \mathbf{y}_{jk^j}) = \frac{1}{1 + D(\mathbf{y}_{ik^i}, \mathbf{y}_{jk^j})} \quad (13)$$

假设概率推理中证据变量个数为 n_E , 对应的证据变量及其取值集合为 $X_{Ek^E} = \{X_{ek^e}\}_{e=1}^{n_E}$, 则证据变量对应的嵌入向量集合为 $\mathbf{y} = \{\mathbf{y}_{ek^e}\}_{e=1}^{n_E}$, 查询变量为 X_q , 取值为 k^q , 对应的嵌入向量为 \mathbf{y}_{qk^q} , m_q 表示查询变量的势. BN 的概率推理就是在给定证据变量的情况下计算查询变量的概率 $P(X_{qk^q} | X_{Ek^E})$. 由于 $P(X_{qk^q} | X_{Ek^E}) = \frac{P(X_{qk^q}, X_{Ek^E})}{P(X_{Ek^E})}$, 根据公式 (13) 计算联合概率 $P(X_{qk^q}, X_{Ek^E})$, 则使用如下方法实现概率推理:

$$P(X_{qk^q} | X_{Ek^E}) = \frac{P(X_{qk^q}, X_{Ek^E})}{P(X_{Ek^E})} = \frac{\sum_{e=1}^{n_E} P(X_{qk^q}, X_{ek^e})}{\sum_{k^q=1}^{m_q} \sum_{e=1}^{n_E} P(X_{qk^q}, X_{ek^e})} = \frac{\sum_{e=1}^{n_E} \text{sim}(\mathbf{y}_{qk^q}, \mathbf{y}_{ek^e})}{\sum_{k^q=1}^{m_q} \sum_{e=1}^{n_E} \text{sim}(\mathbf{y}_{qk^q}, \mathbf{y}_{ek^e})} \quad (14)$$

综上, 基于嵌入向量的概率推理 (autoencoder based Bayesian network embedding and probabilistic inference, ANEI) 思想见算法 2.

算法 2. 基于嵌入向量的概率推理.

输入: $B = (G, \theta)$, BN; $X_{Ek^E} = \{X_{ek^e}\}_{e=1}^{n_E}$, 证据变量集合及其取值; X_{qk^q} , 查询变量 X_q 及其取值 k^q ; \mathbf{Y} , BN 中节点嵌入向量矩阵;

输出: 条件概率 $P(X_{qk^q} | X_{Ek^E})$.

1. $m_q \leftarrow |X_q|$ //得到查询变量 X_q 的势 m_q
2. 从嵌入向量矩阵 \mathbf{Y} 得到证据变量嵌入向量集合 $\{\mathbf{y}_{ek^e}\}_{e=1}^{m_E}$ 及查询变量的嵌入向量 \mathbf{y}_{qk^q}
3. $P(X_{qk^q}, X_{Ek^E}) \leftarrow \sum_{e=1}^{m_E} \frac{1}{1 + D(\mathbf{y}_{ek^e}, \mathbf{y}_{qk^q})}$ //根据公式 (13) 计算证据变量和查询变量的联合概率
4. $P(X_{Ek^E}) \leftarrow \sum_{q=1}^{m_q} P(X_{qk^q}, X_{Ek^E})$ //计算证据变量的边缘分布
5. $P(X_{qk^q} | X_{Ek^E}) \leftarrow \frac{P(X_{qk^q}, X_{Ek^E})}{P(X_{Ek^E})}$
6. **RETURN** $P(X_{qk^q} | X_{Ek^E})$

算法 2 的时间复杂度是 $O(n_E \times m_q)$, 其中 n_E 和 m_q 的大小远远小于 n , 因此算法 2 的时间复杂度远远小于 $O(n^2)$.

4 实验结果

本节使用 5 个不同领域的真实 BN 数据集, 测试基于自编码器的 BN 嵌入方法 (ANE) 和基于嵌入向量的概率推理方法 (ANEI) 的效率和有效性.

4.1 实验数据集

选取用于癌症诊断的 CANCER BN (小型网络)、警报监控系统的 ALARM BN (中型网络)、肝病诊断的 HEPAR2 BN (大型网络)、在线指导学生问题解决的 ANDES BN (超大型网络) 和大型谱系中链接分析的 LINK BN (超大型网络) 这 5 个 BN 数据集, 其统计信息如表 1 所示.

表 1 BN 数据集的统计信息

BN数据集	节点数	边数	参数个数
CANCER	5	4	10
ALARM	37	46	509
HEPAR2	70	123	1453
ANDES	220	338	1157
LINK	724	1125	14211

4.2 对比算法、实验参数及实验指标

4.2.1 对比算法

使用经典的 BN 精确推理方法 VE 和近似推理方法 GS、BNERS 与 ANEI 进行比较.

(1) 变量消元法 (VE): VE 是一种精确推理的概率推理算法, 通过搜索最优的消元顺序逐一进行变量消去, 最终得到概率推理结果^[25]. 本文将基于 VE 的 BN 概率推理结果作为标准结果.

(2) 吉布斯采样 (GS): GS 作为近似推理的一种经典算法, 给定证据变量取值条件下, 根据 BN 中的先验分布采样足够多的样本, 通过统计样本的数量来计算概率推理结果, 采样过程中对采样节点及其马尔可夫覆盖中节点的 CPT 进行重复比较和查询, 拥有指数级别的时间复杂度^[26].

(3) 基于 BN 嵌入的推理 (BNERS): 基于矩阵分解生成 BN 的节点嵌入向量, 保存了 BN 的 DAG 和 CPT, 根据节点嵌入向量之间的相似性来生成对应的样本, 统计样本计算得到近似概率推理结果^[14].

4.2.2 评估指标

将规模不同 BN 下的 ANE 执行时间作为 ANE 效率的评估指标; 为了测试 ANE 的有效性, 比较 ANE 的重构损失函数值与基于矩阵分解的 BN 嵌入的损失函数值, 从而验证 ANE 的有效性.

为了测试 ANEI 的效率, 比较 ANEI 与 VE、GS、BNERS 的多次概率推理的执行时间; 为了测试 ANEI 的有效性, 使用平均绝对误差 (mean absolute error, MAE)、近似度、精度、召回率和 F1 分数来作为评估指标.

(1) 平均绝对误差 (MAE): 用来衡量 ANEI、GS 和 BNERS 概率推理结果和标准结果 (VE) 间的误差.

(2) 近似度: 用来度量 ANEI 比 GS 概率推理结果更接近于标准推理结果的比例. t_{ANEI} 表示 ANEI 推理结果比 GS 更接近于标准结果的次数, t_{GS} 表示 GS 推理结果比 ANEI 更接近于标准结果的次数, 基于 $\frac{t_{\text{ANEI}}}{t_{\text{GS}}}$ 实现近似计算. 近似度的取值范围为 $[0, +\infty]$, 其值为 1 时表示 ANEI 的性能和 GS 相同, 其值大于 1 时表示 ANEI 性能优于 GS, 其值小于 1 时表示 ANEI 性能低于 GS.

(3) 精度 (Precision)、召回率 (Recall) 和 F1 分数 (F1-score): 精度是计算预测为正样本中实际为正样本的概率, 召回率是计算预测实际为正样本中预测为正样本的概率, F1 分数是精度和召回率综合考虑两者的一个度量指标, 预测为真的正样本 (true positive, TP) 表示 ANEI 推理结果与标准值均为正向偏向的推理结果数量, 预测为负的正样本 (false negative, FN) 表示推理结果为负向偏向而标准值为正向偏向的概率推理结果数量, 预测为正的负样本 (false positive, FP) 表示推理结果为正向偏向而标准值为负向偏向的概率推理结果数量.

4.2.3 参数设置

本文使用 Python 3.7 实现所有算法. 通过实验测试将训练迭代次数 *epoch* 设置为 200, 初始学习率设置为 0.01, 嵌入向量维度 *d* 均设置为 10, CANCER BN 和 ALARM BN 中自编码器层数设置为 2; HEPAR2 BN 和 ANDES BN 中自编码器层数设置为 4; LINK BN 中自编码器层数设置为 4.

4.3 实验结果

4.3.1 效率

为了测试概率推理的效率, 对比 VE、GS、BNERs 和 ANEI 在不同 BN 中多次概率推理的执行时间, 以此评估多次概率推理的效率以及 BN 规模对 ANEI 效率的影响; 通过改变 BN 大小、迭代次数 *epoch*、嵌入向量维度 *d*、自编码器层数、损失函数以及是否改变自编码器结构作为主要参数, 记录 ANE 的执行时间, 以此测试 ANE 的效率.

(1) 推理次数对概率推理效率的影响

为了测试基于 ANEI 的多次概率推理效率, 通过改变推理次数 *t* 来比较执行时间, 从图 3(a)–(e) 可以看出: 当在同一个 BN 中进行多次概率推理时, ANEI 的效率均优于其他方法; 在规模最大的 LINK BN 中, 当推理次数大于 5000 时, ANEI 方法比 GS 快近 40 倍. 同时, 为了进一步测试 BN 规模对多次概率推理的效率的影响, 分别从 5 个不同规模的 BN 中随机选择证据变量和查询变量, 对比 ANEI、VE、GS 和 BNERs 在每个 BN 上分别进行 10 次概率推理的执行时间. 从图 3(f) 可以看出, 除 ANEI 推理效率不受 BN 规模大小的影响, 其他概率推理方法都随着 BN 规模的增大而呈指数增长.

(2) BN 大小对 ANE 效率的影响

BN 嵌入过程包括 PMI 矩阵构建前的采样、PMI 矩阵的构建和嵌入, 因此 BN 嵌入总时间 (total time, TT) 由采样时间 (sampling time, ST)、PMI 矩阵构建时间 (PMI time, PT)、嵌入时间 (embedding time, ET) 这 3 部分组成. 为了评估 BN 大小对 ANE 效率的影响, 分别对比 ST、PT、ET 以及 TT, 同时与基于矩阵分解的 BN 嵌入各部分执行时间对比, 如图 4 所示. 可以看出: 在嵌入整个过程中, ANE 和基于矩阵分解的 BN 嵌入方法中 PMI 矩阵的构建具有最大的时间开销、约占 BN 嵌入总时间的 90%; 随着 BN 规模的增大, 节点数量增大, PMI 矩阵构建的时间也就越长, BN 嵌入总时间相应增大.

(3) 参数 *epoch* 对 ANE 效率的影响

为了测试迭代次数 *epoch* 对 ANE 效率的影响, 将 *epoch* 从 10 增加到 500, BN 嵌入时间如图 5(a) 所示. 可以看出, 随着 *epoch* 的增加, ANE 执行时间基本呈线性增加, 且 BN 规模越大, ANE 的效率受 *epoch* 的影响越大, 这因为随着 BN 规模变大, ANE 需要训练的参数越多, 导致 ANE 训练过程的效率越低.

(4) 嵌入向量维度对 ANE 效率影响

为了测试 *d* 对 ANE 效率的影响, 将 *d* 从 2 改变到 50, 统计不同维度下 BN 嵌入执行时间, 如图 5(b) 所示. 结果表明, 随着 *d* 的改变, ANE 的执行时间在同一 BN 中没有明显的差异, 其执行时间几乎相同, 这表明嵌入向量维度 *d* 对 ANE 的效率影响很小.

(5) 自编码器层数对 ANE 效率的影响

为了测试自编码器层数对 ANE 效率的影响, 将层数从 2 层增加到 7 层. 记录规模大小不同的 BN 在不同的编码器层数的条件下所执行的时间, 如图 6 所示. 可以看出, 随着自编码器中层数的增加, ANE 的嵌入时间略微增加, 且增长速度均控制在 10%~20%, 其执行时间基本和自编码器层数成正比.

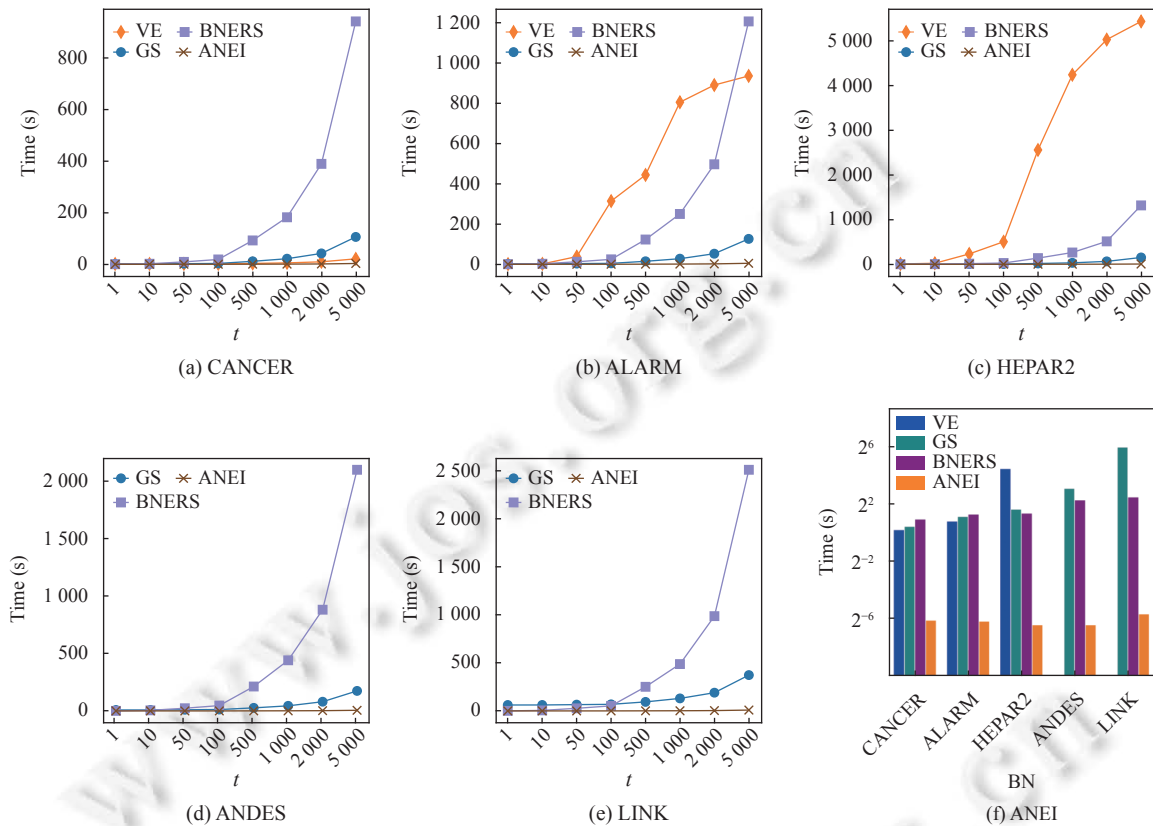


图 3 推理次数 t 对不同 BN 推理效率的影响

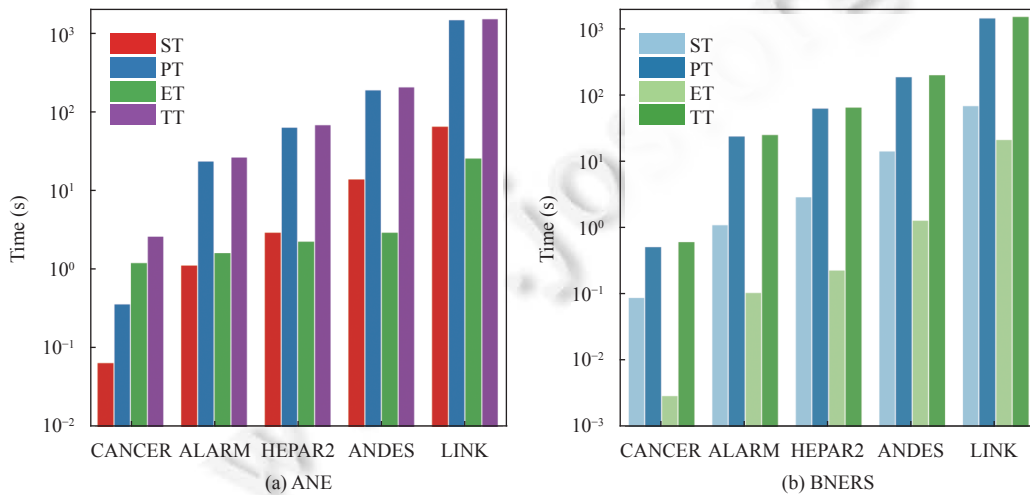


图 4 BN 规模对嵌入时间的影响

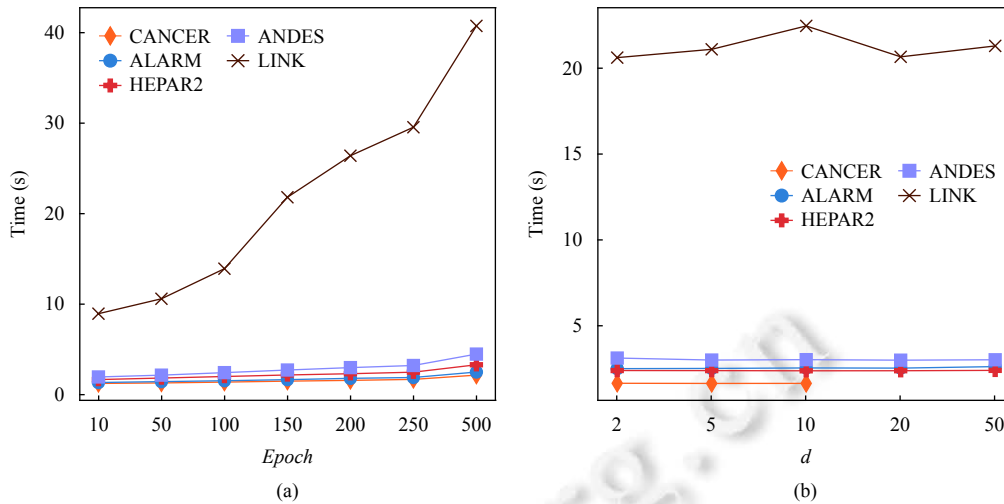
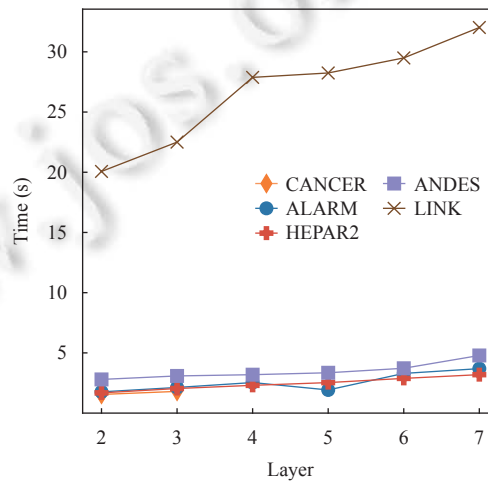
图5 $Epoch$ 和 d 对 ANE 效率的影响

图6 自编码器层数对嵌入时间的影响

(6) 损失函数和改变解码器结构对 ANE 效率的影响

为了测试使用自定义损失函数对 ANE 效率的影响, 记录含有自定义损失函数的 BN 嵌入 (ANE) 和不包含自定义损失函数的 BN 嵌入 (ANE-2st) 执行时间, 如图 7(a) 所示, 可以看出, ANE 的执行时间稍大于 ANE-2st 的执行时间, 这也是因为损失函数的增加导致训练速度变慢, 而在规模最大的 LINK BN 中, ANE 的执行时间比 ANE-2st 的执行时间增加 2%, 意味着 BN 嵌入的效率几乎不受损失函数影响. 同时为了测试改变解码器结构对 ANE 效率的影响, 分别记录了改变解码器结构的 BN 嵌入和不改变解码器结构的 BN 嵌入 (U_ANE) 的执行时间, 如图 7(b) 所示. 可以看出, 自编码器结构的改变也略微增加了 ANE 的执行时间, 这是因为自编码器结构的改变增加了自编码器训练过程中参数, 从增长的时间量来看, 改变解码器结构对 ANE 效率的影响较小.

4.3.2 有效性

为了测试 ANEI 推理结果的有效性, 将基于精确推理算法 VE 得到的推理结果作为对比的标准, 使用 MAE 和近似度来评估概率推理的有效性, 同时通过精度、召回率以及 F1 分数来评估多次概率推理的有效性.

(1) ANEI 有效性

为了验证 ANEI 的有效性, 随机生成查询变量和不同数量的证据变量, 比较 ANEI、BNERS、VE 和 GS 中得

到的推理结果, 结果见表 2. ANEI 的近似度为 1.18, BNERS 的近似度为 0.92, 可以看出, ANEI 和 BNERS 在进行概率推理任务时的表现基本和 GS 相同, 其概率推理结果可以看作合理的概率推理结果.

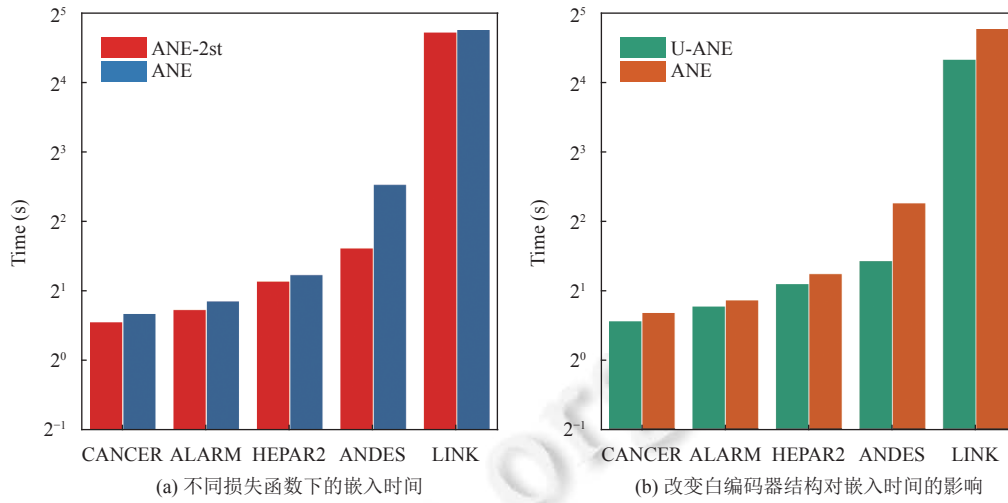


图 7 损失函数变化和自编码器结构改变对 ANE 的影响

表 2 基于 ANEI、BNERS、VE、GS 的随机推理结果

概率推理结果	VE (标准)	GS	BNERS	ANEI	概率推理结果	VE (标准)	GS	BNERS	ANEI	
$P(X_{Q1} X_{e1})$	0.33	0.14	0.45	0.36	$P(X_{Q2} X_{e1}, X_{e2}, \dots, X_{e5}, X_{e6})$	0.40	0.39	0.12	0.43	
$P(X_{Q2} X_{e1})$	0.67	0.86	0.55	0.64	$P(X_{Q1} X_{e1}, X_{e2}, \dots, X_{e6}, X_{e7})$	0.67	0.80	0.59	0.53	
$P(X_{Q1} X_{e1}, X_{e2})$	0.89	0.93	0.56	0.60	$P(X_{Q2} X_{e1}, X_{e2}, \dots, X_{e6}, X_{e7})$	0.33	0.20	0.41	0.47	
$P(X_{Q2} X_{e1}, X_{e2})$	0.11	0.07	0.44	0.40	$P(X_{Q1} X_{e1}, X_{e2}, \dots, X_{e7}, X_{e8})$	0.28	0.01	0.31	0.35	
$P(X_{Q1} X_{e1}, X_{e2}, X_{e3})$	0.86	0.36	0.39	0.44	$P(X_{Q2} X_{e1}, X_{e2}, \dots, X_{e7}, X_{e8})$	0.06	0.04	0.46	0.13	
$P(X_{Q2} X_{e1}, X_{e2}, X_{e3})$	0.03	0.12	0.29	0.31	$P(X_{Q3} X_{e1}, X_{e2}, \dots, X_{e7}, X_{e8})$	0.66	0.95	0.23	0.52	
$P(X_{Q3} X_{e1}, X_{e2}, X_{e3})$	0.11	0.52	0.32	0.25	$P(X_{Q1} X_{e1}, X_{e2}, \dots, X_{e8}, X_{e9})$	0.65	0.76	0.55	0.64	
$P(X_{Q1} X_{e1}, X_{e2}, X_{e3}, X_{e4})$	0.03	0.006	0	0.29	$P(X_{Q2} X_{e1}, X_{e2}, \dots, X_{e8}, X_{e9})$	0.35	0.24	0.45	0.36	
$P(X_{Q2} X_{e1}, X_{e2}, X_{e3}, X_{e4})$	0.97	0.994	1	0.71	$P(X_{Q1} X_{e1}, X_{e2}, \dots, X_{e9}, X_{e10})$	0.30	0.25	0.17	0.25	
$P(X_{Q1} X_{e1}, X_{e2}, X_{e3}, X_{e4}, X_{e5})$	0.61	0.85	0.68	0.54	$P(X_{Q2} X_{e1}, X_{e2}, \dots, X_{e9}, X_{e10})$	0.40	0.44	0.55	0.41	
$P(X_{Q2} X_{e1}, X_{e2}, X_{e3}, X_{e4}, X_{e5})$	0.39	0.15	0.32	0.46	$P(X_{Q3} X_{e1}, X_{e2}, \dots, X_{e9}, X_{e10})$	0.30	0.31	0.28	0.34	
$P(X_{Q1} X_{e1}, X_{e2}, \dots, X_{e5}, X_{e6})$	0.60	0.61	0.88	0.57	与GS相比更接近于VE的次数				11	12

为了进一步测试 ANEI 的有效性, 使用规模最大的 ANDES BN 和 LINK BN、随机选择证据变量和查询变量, 在不同数量的证据变量下分别随机进行 100 次概率推理, 计算 ANEI 和 GS 产生的概率推理结果与标准结果 (VE) 之间的误差, 如表 3 所示. 结果表明, 对于 ANDES 和 LINK 这类规模较大的 BN, ANEI 得到的推理结果和标准结果之间的 MAE 值和 GS 大致相同, 这表明基于 ANEI 的概率推理结果具有和 GS 一样的准确性; 随着证据变量个数的增加, MAE 的变化情况不大, 说明 ANEI 的推理结果对证据变量的数量不敏感, 多次概率推理时准确性不受证据变量数量的限制.

表 3 基于 ANEI 和 GS 随机推理结果的 MAE

BN数据集	1个证据节点		2个证据节点		5个证据节点		9个证据节点	
	ANEI	GS	ANEI	GS	ANEI	GS	ANEI	GS
ANDES	0.199	0.214	0.215	0.199	0.171	0.177	0.195	0.158
LINK	0.309	0.387	0.351	0.295	0.272	0.285	0.327	0.291

(2) 多次概率推理的有效性

为了进一步测试多次概率推理算法的有效性, 分别在 5 个不同规模的 BN 中随机选择证据变量和查询变量, 以 VE 的结果作为标准, 分别计算在不同数量的证据变量的条件下召回率和 $F1$ 分数, 如表 4 所示. 结果表明, 对于不同规模的 BN, 多次概率推理结果的召回率均大于 0.6, $F1$ 分数均大于 0.7; 从 ANEI 与 GS 和 BNERS 的对比结果可以看出, 随着 BN 规模的增大, ANEI 产生的概率推理结果越接近于 GS, 意味着在规模较大的 BN 中 ANEI 的概率推理结果与 GS 类似, 属于有效的近似概率推理方法.

表 4 不同 BN 多次概率推理的召回率和 $F1$ 分数

数据集	指标	算法	1个证据	2个证据	5个证据	9个证据
CANCER	召回率	ANEI	0.62	0.66		
		GS	0.98	0.98		
		BNERS	0.65	0.67		
	$F1$ 分数	ANEI	0.77	0.80		
		GS	0.99	0.99		
		BNERS	0.79	0.80		
ALARM	召回率	ANEI	0.7	0.6	0.72	0.65
		GS	0.78	0.82	0.92	0.90
		BNERS	0.58	0.62	0.67	0.70
	$F1$ 分数	ANEI	0.82	0.75	0.84	0.79
		GS	0.88	0.90	0.96	0.95
		BNERS	0.73	0.77	0.80	0.82
HEPAR2	召回率	ANEI	0.66	0.68	0.72	0.74
		GS	0.96	0.92	0.93	0.92
		BNERS	0.62	0.66	0.68	0.68
	$F1$ 分数	ANEI	0.82	0.75	0.84	0.79
		GS	0.88	0.90	0.96	0.95
		BNERS	0.80	0.81	0.84	0.85
ANDES	召回率	ANEI	0.68	0.64	0.76	0.74
		GS	0.82	0.74	0.70	0.77
		BNERS	0.64	0.68	0.68	0.66
	$F1$ 分数	ANEI	0.81	0.78	0.88	0.85
		GS	0.90	0.85	0.82	0.87
		BNERS	0.78	0.81	0.81	0.80
LINK	召回率	ANEI	0.66	0.64	0.72	0.67
		GS	0.71	0.65	0.73	0.65
		BNERS	0.58	0.60	0.60	0.61
	$F1$ 分数	ANEI	0.80	0.78	0.84	0.80
		GS	0.83	0.78	0.84	0.79
		BNERS	0.73	0.75	0.75	0.77

(3) $epoch$ 对 $F1$ 分数的影响

为了测试自编码器在运行过程中迭代次数 $epoch$ 对 ANEI 概率推理结果有效性的影响, 将 $epoch$ 从 10 逐步改变到 500, 得到的 $F1$ 分数如表 5 所示. 可以看出, $epoch$ 的改变对 $F1$ 分数没有较大的影响, 这是因为本文使用

的 Adam 优化器是一种自适应学习率的优化器, 收敛速度极快, 随着 *epoch* 的增加, 后续用于训练更新的学习率会越来越小, 因此 *epoch* 的值对于后续概率推理的有效性没有明显影响.

(4) 嵌入向量维度对 F1 分数的影响

为了测试 BN 嵌入过程中嵌入维度 d 对 F1 分数的影响, 对于不同规模的 5 个 BN, 将 d 从 2 改变到 50, 其 F1 分数变化如表 6 所示. 可以看出, ANEI 的 F1 分数随着 d 的增加而略有提高, 这表示当 d 逐渐增大时, 嵌入向量包含的 BN 信息越丰富, 表现会更好.

表 5 不同 *epoch* 时的 F1 分数

BN数据集	<i>epoch</i>						
	10	50	100	150	200	250	500
CANCER	0.62	0.62	0.62	0.62	0.62	0.62	0.62
ALARM	0.70	0.70	0.70	0.70	0.70	0.70	0.70
HEPAR2	0.64	0.68	0.62	0.64	0.74	0.70	0.66
ANDES	0.68	0.68	0.68	0.68	0.68	0.68	0.68
LINK	0.64	0.62	0.62	0.60	0.66	0.58	0.52

表 6 不同嵌入向量维度 d 时的 F1 分数

BN数据集	d				
	2	5	10	20	50
CANCER	0.77	0.75	0.77	—	—
ALARM	0.81	0.81	0.82	0.82	0.82
HEPAR2	0.86	0.90	0.92	0.89	0.91
ANDES	0.81	0.81	0.81	0.81	0.81
LINK	0.78	0.77	0.80	0.77	0.75

(5) 自编码器层数对 F1 分数的影响

为了测试自编码器中层数对 F1 分数的影响, 将自编码器的层数从 2 层逐步增加到 7 层, 概率推理中 F1 分数的变化如表 7 所示. 结果表明, 当自编码器层数增加时, BN 的 F1 分数会逐渐增加, 但是随着层数的不断增加, F1 分数也会随之下降, 因此可以确定各 BN 在嵌入过程中编码器的层数, 其中 CANCER BN 取 2 层, ALARM BN 取 3 层, HEPAR2 BN 和 ANDES BN 取 4 层, LINK BN 取 6 层.

(6) 损失函数和改变解码器结构对 F1 分数的影响

为了测试损失函数对多次概率推理有效性的影响, 记录 ANE 和 ANE-2st 的 F1 分数的变化. 如表 8 所示, 结果表明, 在 5 个 BN 中, ANE 的 F1 分数比 ANE-2st 的 ANE 值最高增加了 0.05, 表明我们自定义的损失函数在一定程度上保留了 BN 的原始信息, 有助于优化后续的概率推理结果; 同时为了测试解码器结构改变对后续概率推理准确性的影响, 记录 ANE 和 U-ANE 对 F1 分数的变化, 可以看出, ANE 中 5 个 BN 进行后续概率推理的 F1 分数都相对 U-ANE 提高了 0.05, 表明改变自编码器的结果有助于提升后续推理结果的准确性.

表 7 不同自编码器层数时的 F1 分数

BN数据集	2层	3层	4层	5层	6层	7层
CANCER	0.77	0.77	—	—	—	—
ALARM	0.82	0.84	0.81	0.80	0.81	0.80
HEPAR2	0.75	0.81	0.85	0.84	0.81	0.80
ANDES	0.73	0.81	0.81	0.75	0.77	0.77
LINK	0.73	0.68	0.75	0.75	0.80	0.77

表 8 不同损失函数及不同自编码器结构时的 F1 分数

损失函数	CANCER	ALARM	HEPAR2	ANDES	LINK
ANE-2st	0.73	0.80	0.80	0.81	0.80
U-ANE	0.75	0.80	0.81	0.81	0.75
ANE	0.77	0.82	0.85	0.81	0.80

(7) ANE 的有效性

为了测试 ANE 的有效性, 本文将基于矩阵分解的 BN 嵌入方法和 ANE 的损失函数最终值进行对比, 如表 9 所示. 结果表明: 当 BN 的规模较小时, 基于矩阵分解的重构损失要比 ANE 的重构损失小得多, 而随着 BN 规模的逐渐增大, 基于矩阵分解的重构损失高于 ANE 的重构损失; 从重构损失的值来看, ANE 方法能够有效地保留节点对的 PMI 值, 从而在最终的嵌入向量中保留 BN 的 DAG 和 CPT, 有效保证了后续概率推理的有效性.

为了测试解码器结构的改变对于加快训练速度、降低重构损失值的影响, 分别统计了 ANE 和 U-ANE 在相同迭代次数条件下的重构损失值, 如表 10 所示. 结果表明, 在相同条件下, 改变解码器结构之后, 损失函数值在规模最小的 BN 中减少了 76%, 在规模最大的 BN 中减少了 54%, 意味着改变自编码器结构能够尽可能地降低损失函数值, 从而加快训练速度.

表 9 不同规模 BN 的损失函数值

BN数据集	BNERS	ANE
CANCER	2.81	5.18
ALARM	11.61	26.42
HEPAR2	21.95	35.98
ANDES	41.51	37.65
LINK	282.90	117.48

表 10 是否改变解码器对损失函数值的影响

BN数据集	U-ANE	ANE
CANCER	20.38	5.18
ALARM	163.62	26.42
HEPAR2	66.41	35.98
ANDES	112.43	37.65
LINK	255.75	117.48

综上,实验结果表明,基于嵌入向量的 BN 概率推理方法直接将概率推理转换为嵌入向量之上的高效计算,相比于传统精确推理和近似推理方法,本文提出的概率推理方法效率具有数量级的提升,特别是在大规模 BN 上进行多次概率推理时,本文提出的概率推理方法效率明显高于吉布斯采样,解决了基于矩阵分解的贝叶斯网嵌入的概率推理方法受概率推理次数限制的问题,且推理准确性几乎和经典的吉布斯概率推理方法一样准确.基于自编码器的 BN 嵌入方法可以有效保留节点对的点互信息,从而保存 BN 的有向无环图和条件概率表,且当 BN 规模较大时,本文提出的嵌入方法损失少于基于矩阵分解的 BN 嵌入方法,这为概率推理的有效性提供了保证.

5 总结与展望

本文提出基于自编码器的 BN 嵌入方法和基于嵌入向量的高效概率推理方法,在不同规模 BN 上进行实验,验证了方法的高效性和有效性.随着概率推理次数的增加,本文提出的概率推理方法相比传统的其他概率推理方法效率更高,克服了传统概率推理 NP-难问题以及指数级时间复杂度的瓶颈,适用于需要进行多次概率推理的场景;为了实现高效的 BN 概率推理,本文提出基于自编码器的 BN 嵌入方法,生成的嵌入向量保留了 BN 中节点之间的概率依赖关系,从而保证了推理的有效性.为了保存 BN 中的有向无环图和条件概率表,本文定义了自编码器的解码器结构,并在自编码器损失函数的基础上自定义了新的损失函数,实验结果验证了以上两方面的工作都有较大程度提高了多次概率推理的 F1 分数,即促进了推理结果准确性的提高.

然而,BN 嵌入过程中仍存在信息损失,多次概率推理结果与标准推理结果之间的 MAE 有待提高.因此,未来需要进一步探索 BN 信息损失的下限,对本文的概率推理结果进行优化,进一步提高基于嵌入向量的概率推理方法的准确性.其次,考虑结合知识图谱等知识框架,基于知识图谱等知识框架构建 BN,并利用多次概率推理来预测不同节点之间依赖关系以得到对应的三元组,从知识推理的角度为知识图谱的补全等任务提供支撑.

References:

- [1] Shen Y, Zhang LZ, Zhang J, Yang M, Tang BZ, Li YL, Lei K. CBN: Constructing a clinical Bayesian network based on data from the electronic medical record. *Journal of Biomedical Informatics*, 2018, 88: 1–10. [doi: 10.1016/j.jbi.2018.10.007]
- [2] Yue K, Wu H, Fu XD, Xu J, Yin ZD, Liu WY. A data-intensive approach for discovering user similarities in social behavioral interactions based on the Bayesian network. *Neurocomputing*, 2017, 219: 364–375. [doi: 10.1016/j.neucom.2016.09.042]
- [3] Sun JN, Guo W, Zhang DC, Zhang YX, Regol F, Hu YC, Guo HF, Tang RM, Yuan H, He XQ, Coates M. A framework for recommending accurate and diverse items using Bayesian graph convolutional neural networks. In: *Proc. of the 2020 ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining*. ACM, 2020. 2030–2039. [doi: 10.1145/3394486.3403254]
- [4] Guan SP, Jin XL, Jia YT, Wang YZ, Cheng XQ. Knowledge reasoning over knowledge graph: A survey. *Ruan Jian Xue Bao/Journal of Software*, 2018, 29(10): 2966–2994 (in Chinese with English abstract). <http://www.jos.org.cn/1000-9825/5551.htm> [doi: 10.13328/j.cnki.jos.005551]
- [5] Cooper GF. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 1990, 42(2–3): 393–405. [doi: 10.1016/0004-3702(90)90060-D]
- [6] Dagum P, Luby M. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 1993, 60(1): 141–153. [doi: 10.1016/0004-3702(93)90036-B]
- [7] Qi ZW, Wang JH, Yue K, Qiao SJ, Li J. Methods and applications of graph embedding: A survey. *Acta Electronica Sinica*, 2020, 48(4): 808–818 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2020.04.023]
- [8] Goyal P, Ferrara E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-based Systems*, 2018, 151:

- 78–94. [doi: [10.1016/j.knosys.2018.03.022](https://doi.org/10.1016/j.knosys.2018.03.022)]
- [9] Shi C, Hu BB, Zhao WX, Yu PS. Heterogeneous information network embedding for recommendation. *IEEE Trans. on Knowledge and Data Engineering*, 2019, 31(2): 357–370. [doi: [10.1109/TKDE.2018.2833443](https://doi.org/10.1109/TKDE.2018.2833443)]
- [10] Tu CC, Liu H, Liu ZY, Sun MS. CANE: Context-aware network embedding for relation modeling. In: *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*. Vancouver: ACL, 2017. 1722–1731. [doi: [10.18653/v1/P17-1158](https://doi.org/10.18653/v1/P17-1158)]
- [11] Liao LZ, He XN, Zhang HW, Chua TS. Attributed social network embedding. *IEEE Trans. on Knowledge and Data Engineering*, 2018, 30(12): 2257–2270. [doi: [10.1109/TKDE.2018.2819980](https://doi.org/10.1109/TKDE.2018.2819980)]
- [12] Shen X, Chung FL. Deep network embedding for graph representation learning in signed networks. *IEEE Trans. on Cybernetics*, 2020, 50(4): 1556–1568. [doi: [10.1109/TCYB.2018.2871503](https://doi.org/10.1109/TCYB.2018.2871503)]
- [13] Wang DX, Cui P, Zhu WW. Structural deep network embedding. In: *Proc. of the 2016 ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. San Francisco: ACM, 2016. 1225–1234. [doi: [10.1145/2939672.2939753](https://doi.org/10.1145/2939672.2939753)]
- [14] Qi ZW, Yue K, Duan L, Wang JH, Qiao SJ, Fu XD. Matrix factorization based Bayesian network embedding for efficient probabilistic inferences. *Expert Systems with Applications*, 2021, 169: 114294. [doi: [10.1016/j.eswa.2020.114294](https://doi.org/10.1016/j.eswa.2020.114294)]
- [15] Levy O, Goldberg Y. Neural word embedding as implicit matrix factorization. In: *Proc. of the 27th Int'l Conf. on Neural Information Processing Systems*. Montreal: ACM, 2014. 2177–2185.
- [16] Lee JB, Rossi RA, Kim S, Ahmed NK, Koh E. Attention models in graphs: A survey. *ACM Trans. on Knowledge Discovery from Data*, 2019, 13(6): 62. [doi: [10.1145/3363574](https://doi.org/10.1145/3363574)]
- [17] Perozzi B, Al-Rfou R, Skiena S. DeepWalk: Online learning of social representation. In: *Proc. of the 20th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. New York: ACM, 2014. 701–710. [doi: [10.1145/2623330.2623732](https://doi.org/10.1145/2623330.2623732)]
- [18] Grover A, Leskovec J. Node2vec: Scalable feature learning for networks. In: *Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*. San Francisco: ACM, 2016. 855–864. [doi: [10.1145/2939672.2939754](https://doi.org/10.1145/2939672.2939754)]
- [19] Mahdavi S, Khoshraftar S, An AJ. dynnode2vec: Scalable dynamic network embedding. In: *Proc. of the 2018 IEEE Int'l Conf. on Big Data*. Seattle: IEEE, 2018. 3762–3765. [doi: [10.1109/BigData.2018.8621910](https://doi.org/10.1109/BigData.2018.8621910)]
- [20] Singer U, Guy I, Radinsky K. Node embedding over temporal graphs. In: *Proc. of the 28th Int'l Joint Conf. on Artificial Intelligence*. Macao: IJCAI.org, 2019. 1–8. [doi: [10.24963/ijcai.2019/640](https://doi.org/10.24963/ijcai.2019/640)]
- [21] Cao SS, Lu W, Xu QK. Deep neural networks for learning graph representations. In: *Proc. of the 30th AAAI Conf. on Artificial Intelligence*. Phoenix: ACM, 2016. 1145–1152.
- [22] Park J, Lee M, Chang HJ, Lee K, Choi JY. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In: *Proc. of the 2019 IEEE/CVF Int'l Conf. on Computer Vision*. Seoul: IEEE, 2019. 6519–6528. [doi: [10.1109/ICCV.2019.00662](https://doi.org/10.1109/ICCV.2019.00662)]
- [23] Sankar A, Wu YH, Gou L, Zhang W, Yang H. DySAT: Deep neural representation learning on dynamic graphs via self-attention networks. In: *Proc. of the 13th Int'l Conf. on Web Search and Data Mining*. Houston: ACM, 2020. 519–527. [doi: [10.1145/3336191.3371845](https://doi.org/10.1145/3336191.3371845)]
- [24] Hu FY, Zhu YQ, Wu S, Huang WR, Wang L, Tan TN. GraphAIR: Graph representation learning with neighborhood aggregation and interaction. *Pattern Recognition*, 2021, 112: 107745. [doi: [10.1016/j.patcog.2020.107745](https://doi.org/10.1016/j.patcog.2020.107745)]
- [25] Butz CJ, Oliveira JS, Madsen AL. Bayesian network inference using marginal trees. *Int'l Journal of Approximate Reasoning*, 2016, 68: 127–152. [doi: [10.1016/j.ijar.2015.07.006](https://doi.org/10.1016/j.ijar.2015.07.006)]
- [26] Terenin A, Simpson D, Draper D. Asynchronous Gibbs sampling. In: *Proc. of the 23rd Int'l Conf. on Artificial Intelligence and Statistics*. Palermo: PMLR, 2020. 144–154.

附中文参考文献:

- [4] 官赛萍, 靳小龙, 贾岩涛, 王元卓, 程学旗. 面向知识图谱的知识推理研究进展. *软件学报*, 2018, 29(10): 2966–2994. <http://www.jos.org.cn/1000-9825/5551.htm> [doi: [10.13328/j.cnki.jos.005551](https://doi.org/10.13328/j.cnki.jos.005551)]
- [7] 祁志卫, 王笏辉, 岳昆, 乔少杰, 李劲. 图嵌入方法与应用: 研究综述. *电子学报*, 2020, 48(4): 808–818. [doi: [10.3969/j.issn.0372-2112.2020.04.023](https://doi.org/10.3969/j.issn.0372-2112.2020.04.023)]



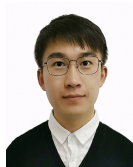
杜斯(1997—), 女, 硕士, 主要研究领域为大数据分析, 不确定性人工智能.



段亮(1986—), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为机器学习, 大数据分析.



祁志卫(1987—), 男, 博士, CCF 学生会员, 主要研究领域为大数据分析, 不确定性人工智能.



王笏辉(1996—), 男, 博士生, 主要研究领域为大数据分析, 不确定性人工智能.



岳昆(1979—), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为大数据分析, 不确定性人工智能.

www.jos.org.cn

www.jos.org.cn