基于动态网络表示的链接预测*

韩忠明[1)2)†] 李胜男[1)] 郑晨烨[1)] 段大高[1)] 杨伟杰[1)]

1) (北京工商大学计算机与信息工程学院，北京 100048)

2) (北京工商大学食品安全大数据技术北京市重点实验室，北京 100048)

(2019 年 7 月 29 日收到; 2020 年 5 月 7 日收到修改稿)

链接预测问题是复杂网络分析领域的重要问题. 现有链接预测方法大多针对静态网络, 忽视了动态信息在网络中的传播. 为此, 针对动态网络中的链接预测问题, 本文提出了一种基于动态网络表示的链接预测 (dynamic network representation based link prediction, DNRLP) 模型. 该模型对网络中不均匀的动态信息进行了学习, 提出了基于连接强度的随机游走算法来模拟动态信息在网络中的扩散, 从而得到新时刻下的节点表示, 然后通过度量节点表示之间的相似度进行链接预测. 实验使用平均交互排序 (mean reciprocal rank, MRR) 和召回率 (*Recall@k*) 指标在四个公开动态网络数据集上进行实验, 结果显示 DNRLP 模型的 MRR 指标较对比模型平均提高了 30.8%. 实验结果表明 DNRLP 模型不仅学习了网络中的动态信息, 还考虑了其对邻居节点的影响以及时间间隔对信息更新的影响, 得到了更为丰富的节点表示, 对于链接预测任务具有明显优势.

关键词： 链接预测, 动态网络, 表示学习, 随机游走

PACS： 89.75.Hc, 29.85.–c, 89.20.Ff **DOI：** 10.7498/aps.69.20191162

1 引 言

在现实世界中, 很多复杂系统以复杂网络的形式出现, 如社会网络、引文网络、生物网络和 web 网络等. 网络提供了一种组织现实世界中的多样化信息的方式, 成为人们工作生活中不可或缺的一部分, 对这些网络进行分析研究具有非常大的学术价值和潜在应用价值[1]. 在这些网络中, 节点之间的交互行为通常以“链接”的形式表示, 即使用边将两个节点连接. 以社交网络为例, 网络节点用于描述用户, 边用于描述用户之间的交互行为. 链接预测[2]通过分析网络中的信息来预测未来网络中任意两个节点之间是否可能出现链接. 有效的链接预测对人们生活中各个方面都具有重要意义, 例如帮助人们控制信息在网络上的传播, 帮助社交平台进行更准确的好友推荐等.

在真实世界中, 网络会随着时间的推移不断进行演变, 即网络中的节点和边会随时间发生变化. 网络演变会导致网络信息发生变化, 进而对链接预测任务产生影响, 因此, 捕获这些网络演化信息是很有必要的. 以社交网络为例, 网络中随时会有新用户注册, 用户随时会创建新的好友关系, 这些新信息的增加不仅改变了当前用户的属性信息, 其邻域的拓扑结构和属性信息也会随之发生改变.

图 1 展示了一个动态网络示意图, 假设在对网络进行链接预测任务时, 以节点共同邻居个数度量节点相似性, 相似度越大的节点对在下一时刻发生链接的可能性越大. 在 T1 时刻, 网络中的节点 2 和节点 5 拥有一个共同邻居 (节点 4), 在 T2 时刻, 该网络在节点 3 和节点 5 之间新增了一条边, 即节点 3 变成了节点 5 的邻居. 此时节点 2 和节

* 北京市自然科学基金 (批准号: 4172016) 和北京市哲学社会科学基金一般项目 (批准号: 14ZHB006) 资助的课题.

† 通信作者. E-mail: hanzhongming@btbu.edu.cn

点 5 拥有两个共同邻居 (节点 4 和节点 3), 它们在下一时刻产生链接的可能性变大. 由此可见, 虽然新增加的边只涉及到节点 3 和节点 5, 但其邻域中的节点 2 的属性也受到了影响. 因此, 网络动态演化对节点及其邻域的特征信息有着非常重要的影响, 在链接预测过程中加入动态信息将会提高链接预测的性能.

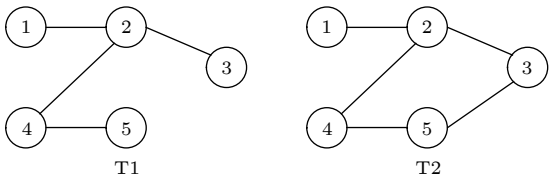

图 1 动态网络示意图

Fig. 1. Schematic diagram of dynamic network.

现有的链接预测方法大多针对静态网络, 使用网络拓扑结构特征分析的方法进行链接预测, 当网络信息发生变化时, 其性能将会受到很大影响. 此外, 网络中的节点并不是每时每刻都在产生新的交互信息, 其发生变化的时间是不规律的, 即变化发生的时间分布不均匀. 而两次变化之间的时间间隔会影响节点的偏好信息. 例如, 如果某节点两次变化之间的时间间隔较长, 则应该更关注新的交互信息, 因为新的信息更能体现该节点当前的偏好. 为了有效地捕获网络中的动态演化信息, 本文使用表示学习方法, 用低维稠密向量表示网络节点的偏好信息, 通过度量网络节点表示的相似度进行链接预测, 并提出了基于动态网络表示的链接预测模型 DNRLP(dynamic network representation based link prediction). 针对网络演化产生的动态信息, DNRLP 设计了基于不均匀时间间隔的信息更新机制. 同时, 考虑到动态信息对相关节点邻域的影响, 设计了基于连接强度的随机游走算法对邻域信息进行更新. 该模型可以有效地捕获网络动态信息, 提高链接预测的质量和有效性.

2 相关研究

2.1 链接预测

现有的链接预测研究方法主要分为两类, 基于网络拓扑结构特征分析的方法和基于机器学习的方法. 传统的链接预测方法主要是通过对网络拓扑结构进行特征分析, 计算节点之间的相似度, 认为相似度高的节点在将来会发生链接. Newman 等[3] 首先提出基于网络共同邻居的节点相似度计算方法, 即节点拥有的共同邻居越多, 越可能在未来发生链接. Adamic 等[4] 提出了一种新的网络节点相似性度量方法, 该方法根据共同邻居节点的链接情况为每个邻居节点设置权重, 并使用其加权和作为节点对的相似度. Fouss 等[5] 通过随机游走算法对网络中节点的邻域信息进行采样, 得到目标节点的随机游走序列, 然后计算节点随机游走序列的相似性进行链接预测.

随着人工智能和机器学习技术的快速发展, 越来越多的研究人员尝试使用机器学习方法去解决链接预测问题. 基于机器学习的链接预测方法首先需要从网络中得到各个节点的特征向量, 然后将节点的特征向量作为机器学习算法的输入. Hasan 等[6] 将链接预测问题转化为机器学习中的二分类问题, 尝试使用支持向量机[7], 多层感知机等机器学习方法进行链接预测, 若两节点间未来可能产生链接则预测值为 1, 否则为 0. Freno 等[8] 使用自然语言处理领域的词袋模型对论文引用网络中论文的摘要进行建模, 得到论文节点的特征表示, 然后使用神经网络进行链接预测. Hosein 等[9] 针对引文网络使用论文作者和论文的互聚类方法进行链接预测. Xu 等[10] 将信息熵应用于加权网络中的链接预测, 提出基于路径贡献的加权相似度指标, 实现了加权网络的链接预测. Lai 等[11] 针对复杂网络, 用模块化的置信度传播算法来获得网络的底层块结构, 并通过块结构信息对节点间产生链接的可能性进行建模, 从而实现链接预测. Kovács 等[12] 针对蛋白质相互作用网络, 根据蛋白质之间的交互特性, 使用长度为 3 的网络路径 (L3) 进行链接预测. Pech 等[13] 提出了一种新的链接方法, 由节点邻居贡献率的线性和来估计链接的可能性, 从而将链接预测问题转化为似然矩阵的优化问题. Zhang 等[14] 认为现有的相似性度量方法往往只适用于某几种网络, 为此提出了一种 γ-衰减理论来统一现有的相似性度量方法, 同时还提出了一种基于图神经网络 (graph neural network, GNN)[15] 的链接预测框架 SEAL, 从网络中的局部子图来学习节点表示以进行链接预测. 以上方法大多是针对特定网络提出了新的相似性度量方法. 除此之外, Ostapuk 等[16] 首次将深度主动学习[17] 应用于链接预测, 基于贝叶斯深度学习[18] 提出了一种深度主动学习框架 ActiveLink, 将不确定性采样引入到聚类算法中,

并且采用基于元学习[19]的无偏增量的方法进行训练, 提高了模型的训练速率. 相较于传统的基于网络结构相似度的链接预测方法而言, 有监督的机器学习模型使链接预测的结果有了明显提升.

由于真实世界中的网络是随时间不断演化的, 因此虽然上述方法在大规模网络的链接预测中取得了较好的成果, 但其大多仅考虑了网络结构且大多只适用于静态网络, 而忽视了真实网络中的动态信息以及节点间产生链接的时间信息, 因而在网络发生变化时需要进行大量的重新计算.

2.2 动态网络表示学习

由于复杂网络通常包含数十亿的节点和边, 且数据具有稀疏性, 在网络上很难直接进行复杂的推理过程, 为了有效地进行复杂网络分析, 学者们提出了各种各样的网络表示学习[20]方法. 网络表示学习作为网络分析领域的一个重要基础问题, 其核心思想是寻找一个映射函数将网络中的节点转化成低维稠密的实数向量, 即网络节点表示. 这些网络节点表示保存了网络中所包含的信息, 为网络分析任务提供了良好的特征基础, 并可以直接用于各种网络分析任务中, 如链接预测, 社团检验, 推荐系统等. 网络表示学习的形式化定义如下:

对于给定网络 $G=(V,E)$, 使用映射函数 $f_v \to \tau^k$ 为网络中的每个节点 $v \in V$ 学习到一个低维稠密的实数向量 $\boldsymbol{R}_v \in \mathbb{R}^k$ 作为节点的表示向量, 该向量的维度 k 远远小于网络节点的总个数 $|V|$.

由于网络表示在常见的网络分析任务中展现出了良好的能力, 因此越来越多的学者关注于网络表示学习领域, 并提出了多种网络表示学习方法, 如 DeepWalk[21], LINE[22], node2vec[23], SDNE[24], GCN[25], GraphSAGE[26] 等.

近年来, 针对动态网络的表示学习研究逐渐受到了研究人员的关注. 如 Michael 等[27]基于复杂网络动力学以及多元微分方程定义节点在不同时刻的表示, 提出了一种复杂网络的多尺度动态嵌入技术. Kumar 等[28]基于递归神经网络提出了 JODIE 模型, 对网络中的用户和项目分别进行动态表示学习, 并提出了一种并行批处理算法 t-Batch. 李志宇等[29]通过对不同阶层的网络节点关系进行正负阻尼采样, 构建针对新增节点的动态特征学习方法, 使得模型可以提取大规模社会网络在动态变化过程中的结构特征. Palash 等[30]基于深度自编码器提出 DynGEM 模型, 该模型可以动态学习网络中高度非线性的表示. 同时很多学者针对动态网络表示学习中的链接预测任务进行了相关研究. Chen 等[31]将长短期记忆网络[32](LSTM) 与编码器-解码器体系结构相结合, 提出了一种新颖的 encoder-LSTM-decoder(E-LSTM-D) 深度学习模型来预测动态链接. Li 等[33]基于 SDNE 算法提出了 DDNE 模型, 使用门控循环单元[34](GRU) 作为编码器来捕获动态网络中的时间信息, 从而在动态网络中进行链接预测. Lei 等[35]结合了图卷积网络 (graph convolutional network, GCN)、长短期记忆网络 (long short-term memory, LSTM) 以及生成对抗网络[36](generative adversarial networks, GAN) 的优势, 用深度神经网络 (即 GCN 和 LSTM) 来探索网络中隐藏的拓扑结构和演化模式的非线性特征, 用 GAN 来解决动态网络中链接的稀疏性问题, 同时通过对抗的方式在动态网络中进行链接预测. 这些研究方法大多只考虑了发生变化的节点本身的信息变化情况, 而没有关注节点邻域所受到的影响. 并且现有方法大多仅考虑了均匀间隔的时间间隔, 而忽视了不同时间间隔对节点偏好信息的影响. 由于网络表示学习是网络分析的基础任务, 如何设计具有动态适应性的网络表示学习模型, 学习网络节点及其邻域的信息变化并对它们的表示进行快速更新, 对现实世界中的网络分析任务有着至关重要的作用.

3 基于动态网络表示的链接预测模型

本文针对动态网络的链接预测问题提出了基于动态网络表示的链接预测模型 DNRLP. 该模型对 LSTM 进行了改进, 考虑了网络演化过程中产生新信息的非平均时间间隔问题以及新信息的扩散问题, 有效地捕获和学习了网络中的动态信息, 并得到了含有节点偏好信息的节点表示. 然后通过计算习得节点表示之间的相似度, 最终得到链接预测的结果.

图 2 给出 DNRLP 模型的结构示意图, DNRLP 模型主要分为两个模块: 动态网络表示学习模块和链接预测模块, 其中动态网络表示学习模块由节点信息动态更新单元和节点邻域更新单元组成. DNRLP 模型根据 T_i 时刻网络中出现的新增信息, 得到与其直接关联的节点集合, 使用节点信息动态

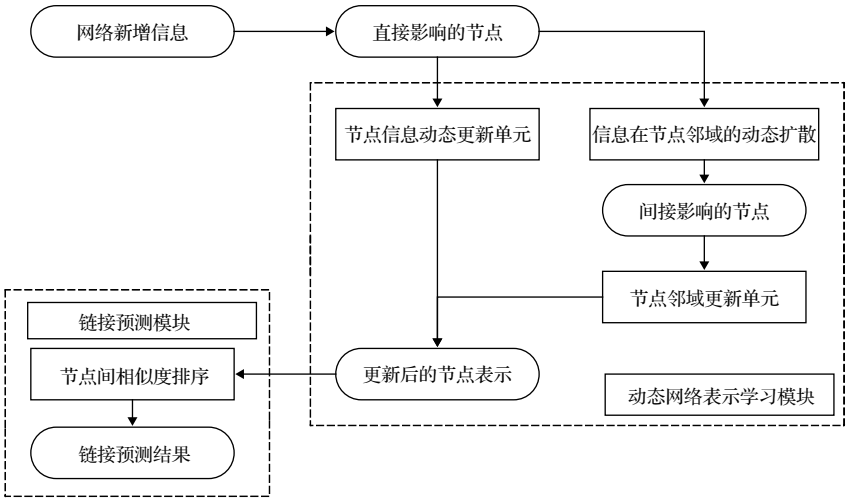

图 2 基于动态网络表示的链接预测模型结构

Fig. 2. The architecture of link prediction model based on dynamic network representation.

更新单元对该集合内的节点进行节点表示更新. 然后对该集合内的节点进行邻域采样, 得到与新增信息间接关联的节点集合, 使用节点邻域更新单元对邻域节点进行更新, 最终得到当前时刻更新后的网络节点表示. 基于这些节点表示使用链接预测模块计算节点间的相似度并进行排序, 最终得到链接预测的结果.

3.1 节点信息动态更新

随时间动态演化的网络可以看作不同时刻下的静态网络, 使用 $G(V^t, E^t, t)$ 表示 t 时刻的网络, 其中 V^t 为该时刻的节点集合, E^t 为该时刻的边集合, t 为对应的时间戳. 随着时间的推移, 网络中的节点会不断地与网络中的其他节点建立新链接, 这些新链接会改变当前节点的属性信息. 例如在社交网络中, 如果两个用户有联系, 他们会逐渐分享共同的兴趣爱好. 新链接的建立顺序以及它们建立的时间间隔对节点属性特征的变化也有着非常重要的影响. 按照时间戳对节点 v 新产生的链接进行排序得到链接序列 $S_v = \{(v, v_i, t_0), (v, v_i, t_1), \ldots, (v, v_i, t_n)\}$, 其中 (v, v_i, t) 表示 t 时刻节点 v 与节点 v_i 之间新建立的链接, $v_i \in N_v$ 表示节点 v 的一阶邻域节点, N_v 表示节点 v 的一阶邻域节点集合; t 表示链接建立的时间戳, $t_0 < t_1 < \cdots < t_n$. 在链接序列 S_v 中, 链接建立的时间 t 越晚, 链接的排序越靠后, 则对节点 v 属性变化的影响越大. 新链接之间的时间间隔 Δt 即链接序列 S_v 里两个相邻新链接 $((v, v_i, t-x)$ 与 $(v, v_i, t))$ 之间的时间戳之差的绝对值, $\Delta t = |t - (t-x)|$. 其形式化定义如下: 给定一个链接序列 $S_v = \{(v, v_i, t_0), (v, v_i, t_1), \ldots, (v, v_i, t_n)\}$, 新链接之间的时间间隔 Δt 定义为: 在链接序列 S_v 中, 链接 $(v, v_i, t-x)$ 建立的时间戳 $t-x$ 与其后一个链接 (v, v_i, t) 建立的时间戳 t 的差的绝对值 $|t-(t-x)|$, 即 $\Delta t = |t-(t-x)|$. 时间间隔 Δt 的值越大则次序较后的链接对节点属性变化的影响越大. 如在社交网络中, 用户的新增关注可能呈现了该用户最新的偏好, 而时间较久远的关注对用户产生的影响会随着时间的推移不断减小. 因此, 节点建立的两个链接之间的时间间隔将会影响节点表示向量的更新策略. 如果节点两次建立链接之间的时间间隔较大, 那么模型应更关注新链接所带来的信息, 而对时间较久远的信息进行遗忘.

综上所述, 针对网络中的任一节点, 当产生新链接时, 应该根据链接产生的时间间隔决定需要更新哪些新信息, 以及需要遗忘哪些历史信息. DNRLP 模型基于 LSTM 模型对网络中的节点进行动态表示学习. LSTM 模型通过遗忘门、输入门和输出门解决了对历史信息的长期依赖问题. 但是现有的 LSTM 中没有考虑不同的时间间隔对历史信息丢弃策略所产生的影响, 因此我们根据动态网络信息传播规律, 在 LSTM 的计算过程中增加了一个基于时间间隔的信息过滤单元 (time interval based filter unit, TIFU), 从而达到了根据时间间隔的大小决定下一时刻节点对历史信息的丢弃程度的目的, 使模型更关注节点的新增信息, 其计算单元如图 3 所示.

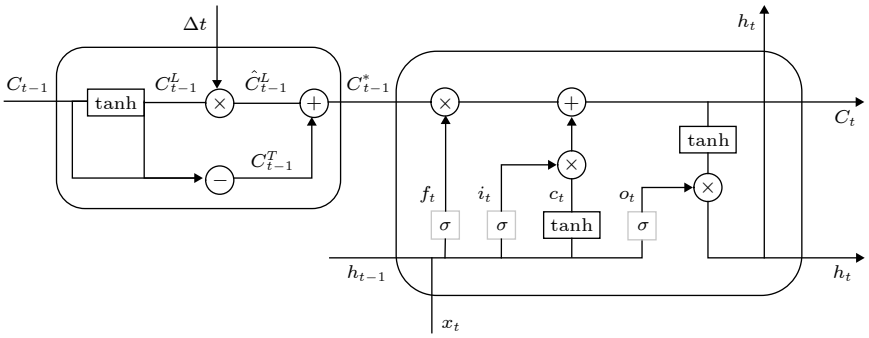

图 3 基于时间间隔的 LSTM 单元

Fig. 3. Time interval based LSTM unit.

图 3 左半部分描述了 TIFU 的示意图. TIFU 的工作原理是根据时间间隔 Δt 的大小, 决定当前细胞状态信息 $\boldsymbol{C}_{t-1}$ 传递到下一时刻 t 的信息 $\boldsymbol{C}^*_{t-1}$. $\boldsymbol{C}^*_{t-1}$ 的具体计算过程如下所示:

$$\boldsymbol{C}^L_{t-1} = \tanh\left(\boldsymbol{W}_d\boldsymbol{C}_{t-1} + \boldsymbol{b}_d\right), \tag{1}$$

$$\boldsymbol{C}^T_{t-1} = \boldsymbol{C}_{t-1} - \boldsymbol{C}^L_{t-1}, \tag{2}$$

$$\widehat{\boldsymbol{C}}^L_{t-1} = \boldsymbol{C}^L_{t-1} * \left(1/\log(\mathrm{e} + \Delta t)\right), \tag{3}$$

$$\boldsymbol{C}^*_{t-1} = \boldsymbol{C}^T_{t-1} + \widehat{\boldsymbol{C}}^L_{t-1}, \tag{4}$$

上述公式中, TIFU 将上一时刻 $t-1$ 标准 LSTM 计算单元输出的细胞状态 $\boldsymbol{C}_{t-1}$ 分成了两个部分: 短期记忆 $\boldsymbol{C}^L_{t-1}$ 和长期记忆 $\boldsymbol{C}^T_{t-1}$. 我们认为细胞状态 $\boldsymbol{C}_{t-1}$ 是由长期记忆和短期记忆两个部分构成的, 短期记忆对信息的存储时间较短, 容易被遗忘, 而长期记忆对信息的存储时间较长, 不容易被遗忘. 同时短期记忆与长期记忆并不是完全割裂的, 通过重复、巩固短期记忆可以转化为长期记忆, 即随着时间的流逝, 部分短期记忆可以演变为长期记忆. (1) 式使用神经网络和 tanh 激活函数自动选择历史信息中较为短暂的历史信息, 即单元的短期记忆, 其中 $\boldsymbol{C}^L_{t-1}$ 为根据 $t-1$ 时刻的细胞状态生成的短期记忆. (2) 式中 $\boldsymbol{C}^T_{t-1}$ 为相应的需要传递给下一时刻 t 的长期记忆. TIFU 根据时间间隔 Δt 对单元短期记忆 $\boldsymbol{C}^L_{t-1}$ 的部分信息进行丢弃, 如 (3) 式所示, 其中 $\widehat{\boldsymbol{C}}^L_{t-1}$ 为保留下来的短期记忆信息, Δt 越大丢弃的短期记忆信息越多. 经过上述计算, 完成对节点历史信息保留的决策, 并得到需要传递给下一时刻 t 的历史信息, 如 (4) 式所示, $\boldsymbol{C}^*_{t-1}$ 将 $\boldsymbol{C}^T_{t-1}$ 和 $\widehat{\boldsymbol{C}}^L_{t-1}$ 进行组合, 并作为下一时刻 t 标准 LSTM 单元的输入, 即最终传递给下一时刻 t 的节点历史信息是由节点的部分短期记忆与全部长期记忆所组成的.

图 3 中右半部分为标准 LSTM 计算单元示意图, 其具体计算过程如下所示:

$$\boldsymbol{x}_t = \boldsymbol{W}_1\boldsymbol{u}_{v_i} + \boldsymbol{W}_2\boldsymbol{u}_{v_j} + \boldsymbol{b}, \tag{5}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i\boldsymbol{h}_{t-1} + \boldsymbol{U}_i\boldsymbol{x}_t + \boldsymbol{b}_i), \tag{6}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f\boldsymbol{h}_{t-1} + \boldsymbol{U}_f\boldsymbol{x}_t + \boldsymbol{b}_f), \tag{7}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o\boldsymbol{h}_{t-1} + \boldsymbol{U}_o\boldsymbol{x}_t + \boldsymbol{b}_o), \tag{8}$$

$$\tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_c\boldsymbol{h}_{t-1} + \boldsymbol{U}_c\boldsymbol{x}_t + \boldsymbol{b}_c) \tag{9}$$

$$\boldsymbol{c}_t = f_t \odot \boldsymbol{C}^*_{t-1} + i_t \odot \tilde{\boldsymbol{c}}_t, \tag{10}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{c}_t\right) \tag{11}$$

其中 $\boldsymbol{x}_t$ 为当前时刻 t 的输入向量, 表示网络的新增信息. 由于新增信息由节点 v_i, v_j 之间的新增链接产生, 因此可以通过计算两节点当前表示的加权和来得到 $\boldsymbol{x}_t$, 计算方式如 (5) 式所示. 接下来分别对标准 LSTM 单元的输入门、遗忘门及输出门进行计算, 其中 σ 表示 sigmoid 激活函数, $\odot$ 表示矩阵乘积运算, $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$ 分别代表 t 时刻 LSTM 单元输入门、遗忘门以及输出门的系数. $\{\boldsymbol{W}_i, \boldsymbol{U}_i, \boldsymbol{b}_i\}$, $\{\boldsymbol{W}_f, \boldsymbol{U}_f, \boldsymbol{b}_f\}$ 和 $\{\boldsymbol{W}_o, \boldsymbol{U}_o, \boldsymbol{b}_o\}$ 分为上述三种门的网络参数. $\tilde{\boldsymbol{c}}_t$ 表示用于更新细胞状态 $\boldsymbol{c}_t$ 的候选状态. $\{\boldsymbol{W}_c, \boldsymbol{U}_c, \boldsymbol{b}_c\}$ 是网络产生候选记忆的参数. $\boldsymbol{h}_t$ 是在时刻 t 时经过上述三种门的过滤后的隐藏状态, 该状态记录了 t 时刻之前习得的所有有用信息. $\boldsymbol{c}_t$ 经过输出门舍弃掉部分信息后形成当前时刻 t 的输出向量 $\boldsymbol{h}_t$. 根据上述 TIFU 和标准 LSTM 计算单元的计算过程, 可将上述过程进行如下表示:

$$\boldsymbol{C}_t, \boldsymbol{h}_t = f(\boldsymbol{C}_{t-1}, \boldsymbol{h}_{t-1}, \boldsymbol{x}_t, \Delta t), \tag{12}$$

当网络中有新信息产生时, 使用 f 对关系两端的节点信息 (节点表示) 进行更新, 其中 $\boldsymbol{C}_{t-1}$, $\boldsymbol{h}_{t-1}$ 为上一时刻 f 计算得到的细胞状态和隐藏状

态, $\boldsymbol{x}_t = \boldsymbol{W}_1\boldsymbol{u}_{v_i} + \boldsymbol{W}_2\boldsymbol{u}_{v_j} + \boldsymbol{b}$是网络新增关系为涉及到的两个节点$\boldsymbol{v}_i$和$\boldsymbol{v}_j$带来的新信息, $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}$是生成新信息的表示向量的模型参数. $\boldsymbol{h}_t$即为目标节点更新后的表示向量.

针对模型冷启动问题, 在初始时刻, DNRLP模型使用网络的邻接矩阵作为网络节点的表示向量, 并对网络中每个节点进行固定大小的邻域采样, 然后使用聚合函数对节点邻域内的节点表示进行聚合, 最终得到节点初始化表示向量, 并使用上述表示向量作为f的初始化的节点表示.

3.2 节点信息扩散算法和更新

网络中两节点v_i, v_j之间的新增链接不仅会对链接两端的节点产生影响, 同时也会影响与v_i, v_j距离较近的节点. 因此当网络产生新链接时, 涉及到的两个节点v_i, v_j的邻域节点也应该进行信息更新. 为此, DNRLP模型通过对产生新链接的节点进行邻域采样来模拟新信息在网络中的扩散过程, 然后对采样到的邻域节点进行信息更新. 这么做的原因主要有三个方面: 第一, 文献[37]表明新链接对整个网络的影响往往是局部的. 第二, 由于网络的复杂性, 与新链接直接关联的节点不一定会将收集到的新信息传播给其所有的邻居, 同时新信息很有可能会被传播到与其较近但不直接相邻的节点. 第三, 通过实验发现, 当对目标节点的局部邻域进行信息更新时, 模型的性能会更好.

在节点邻域采样的过程中, DNRLP模型采用基于连接强度的随机游走算法. 把节点间的连接强度作为随机游走中的边权重, 对目标节点进行加权随机游走采样从而得到节点v_i, v_j的局部邻域. 其中边权重的计算过程如下:

$$f_{\mathrm{s}}\left(\boldsymbol{u}_{v_i}, \boldsymbol{u}_v\right) = \frac{\exp(\boldsymbol{u}_{v_i} \cdot \boldsymbol{u}_v)}{\sum_{v_i \in N_v} \exp(\boldsymbol{u}_{v_i} \cdot \boldsymbol{u}_v)}, \tag{13}$$

其中, $\boldsymbol{u}_v$为节点v的表示向量, N_v表示节点v的一阶邻居节点集合, $f_{\mathrm{s}}\left(\boldsymbol{u}_{v_i}, \boldsymbol{u}_v\right)$表示节点$v$和其邻域节点$v_i$间的连接强度, 可以将该连接强度看作一个归一化后的概率值, 根据该概率值来选择目标节点信息在下一时刻要扩散到的节点. 图4给出一个简单网络实例, 图中实线代表历史链接, 虚线代表当前时刻新产生的链接. 分别对网络中新链接两端的节点v_4, v_5进行随机游走. 以节点v_5的随机游走邻域采样为例, 其具体的随机游走采样策略如下:

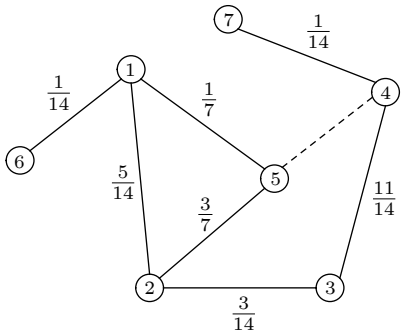

图 4 节点邻域采样示意图

Fig. 4. Schematic diagram of node neighborhood sampling.

(1) 建立随机游走结果集合$R_{v_5} = \{\}$.

(2) 根据边权重概率分布随机选择下一节点v_1, 并将该节点加入R_{v_5}中.

(3) 判断所选节点v_1是否有一阶邻居, 或者其一阶邻居是否全部在R_{v_5}中, 是则退回到上一时刻的节点重复此步骤, 否则进入下一步.

(4) 重复步骤(2)和(3)直到随机游走的结果集合达到期望的长度.

(5) 若随机游走过程中选择的节点与结果集合中的节点重复, 则退回到上一时刻的节点重新进行选择. 如图中节点v_2下一刻游走选择节点v_5, 则退回到节点v_2重新进行决策.

表1给出了扩散算法的伪代码. 在表1中, E_{new}代表新增链接的集合; v代表与新增链接相关

表 1 信息扩散算法

Table 1. Information diffusion algorithm.

输入: 新增链接$e_{ij} \in E_{\mathrm{new}}$, 随机游走长度$L$

输出: 随机游走序列R

1) For e_{ij} in E_{new} do:
2) For v in e_{ij} do:
3) $m = 0$
4) While $m < L$ do
5) 初始化权重分布P
6) For u in N_v do
7) 根据(13)式计算$f_{\mathrm{s}}\left(u_{\mathrm{u}}, u_v\right)$, 加入$P$
8) End for
9) 根据P选择下一个节点u'加入R_v
10) $m = m + 1$
11) $v = u'$
12) End while
13) 将R_v加入R
14) End for
15) End for

联的一个节点; m代表随机游走了的长度; L是给定的随机游走序列的最大长度; P表示边权重概率分布; u代表节点v的一阶邻居; R_v代表节点v的随机游走结果集合; R代表所有节点的随机游走结果集合. 步骤 6—8 实现节点间边权重的计算. 步骤 9 实现相关节点的邻域采样. 步骤 4—12 实现基于连接强度的随机游走算法, 找到了相关节点的局部邻域 R_v, 其中 R_v是一个有序的随机游走序列, 越靠前的节点越容易从相关节点到达, 即相关节点的信息更容易扩散到序列中排位靠前的节点上去, 刻画出了相关节点信息的扩散过程. 整个算法得到了与新增信息直接相关的节点的随机游走序列 R_v 的集合 R, 描绘出了整个网络中新增信息的扩散过程.

由于新增链接并没有对随机游走序列中的节点产生直接影响, 因此新增链接的信息并没有影响这些节点的历史信息, 只带来了新信息, 并且对于相关节点局部邻域中较老、较远的节点 (较老的节点: 相关节点与其的交互发生在比较早先的时候或者其与随机游走序列中的上一个节点之间的交互发生在比较早先的时候; 较远的节点: 相关节点与其的距离比较远) 而言, 新信息对其影响较小. 综上, DNRLP 模型根据相关节点与其随机游走序列中的节点之间的链接存在的时间长短, 或者根据随机游走序列中相邻两节点之间链接存在的时间长短, 对新信息进行处理. 建立链接的时间越长, 需要丢弃的新增信息越多. 同时还使用相关节点与其随机游走序列中节点之间的距离对新信息进行进一步的处理.

DNRLP 模型设计了节点邻域更新单元对新信息涉及到的相关节点 v_i 的随机游走序列进行信息更新, 更新过程如下:

$$\boldsymbol{C}_v^t = \boldsymbol{C}_v^{t-1} + 1/ind_v * (1/\log(\mathrm{e}+\Delta t)) * \boldsymbol{W}_s * \boldsymbol{x}_t \quad (14)$$

$$\boldsymbol{h}_v^t = \tanh\left(\boldsymbol{C}_v^t\right) \quad (15)$$

式中, $v \in R_v$, ind_v为节点v在R_v中的索引号, $\boldsymbol{C}_v^{t-1}$为节点v上一时刻的细胞状态, $\boldsymbol{x}_t = \boldsymbol{W}_1 \boldsymbol{u}_{v_i} + \boldsymbol{W}_2 \boldsymbol{u}_{v_j} + \boldsymbol{b}$为节点$v_i$和$v_j$之间新增关系产生的新信息. 更新后节点$v$的表示向量为$\boldsymbol{h}_v^t$. 上述相关节点邻域信息更新单元的结构如图 5 所示.

3.3 参数训练

为了在无监督方式下进行参数学习, DNRLP 模型将输出的节点表示向量$\boldsymbol{h}_v$, $v \in V$应用于基于图的损失函数, 并使用梯度下降法对模型参数进行更新. 基于图的损失函数假设相互连接的节点有着相似的网络表示向量, 损失函数如下:

$$J(\boldsymbol{h}_v) = -\log\left[\sigma\left(\boldsymbol{h}_v^{\mathrm{T}} \boldsymbol{h}_u\right)\right] - Q \cdot E_{v_n \sim p_{n(v)}} \log[-\sigma(\boldsymbol{h}_v^{\mathrm{T}} \boldsymbol{h}_{v_n})] \quad (16)$$

其中, $\sigma\left(h_v^T h_u\right)$定义了节点$v$和节点$u$之间存在链接的概率, $p_{n(v)}$为负采样分布, Q定义了负采样的数量. 通过该损失函数习得的网络表示包含了网络节点之间的交互信息, 可以直接用于后续的链接预测任务.

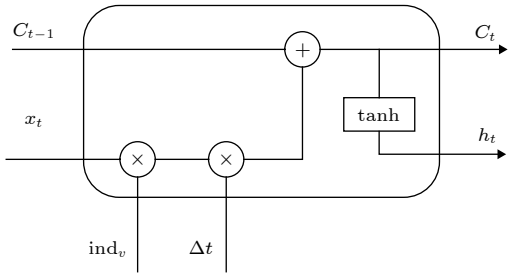

图 5 节点邻域更新单元

Fig. 5. Node neighborhood update unit.

3.4 基于动态网络表示的链接预测

在网络中相似节点在未来发生链接的可能性更大, 因此, 本文通过度量网络节点之间的相似度来进行网络链接预测. 通过上述动态网络表示学习过程, 我们可以得到每次网络演化后的新节点偏好表示, 这些节点表示保存了节点的偏好信息, 可以直接进行节点间的相似度计算, 计算过程如下:

$$sim(v,u) = \frac{\sum_{i=1}^{n} \boldsymbol{h}_v^i \times \boldsymbol{h}_u^i}{\sqrt{\sum_{i=1}^{n} \left(\boldsymbol{h}_v^i\right)^2} \cdot \sqrt{\sum_{j=1}^{n} \left(\boldsymbol{h}_u^j\right)^2}} \quad (17)$$

其中, $\boldsymbol{h}_v$和$\boldsymbol{h}_u$表示两个节点在当前时刻的表示向量, i和j表示节点偏好表示向量的分量. 相似度越大, 则节点间发生链接的可能性越大, 因此对网络目标节点进行链接预测时, DNRLP 模型首先会计算该节点与网络中的其余节点之间的相似度并对其进行排序, 选择 top-k 的节点作为最终链接预测的结果.

4 实验分析

4.1 实验设计

为了验证 DNRLP 模型在网络链接预测任务下的性能和有效性, 本文在具有代表性的四个公开动态网络数据集上进行了对比实验. 这四个数据集的数据统计信息如表 2 所示.

表 2 动态网络数据详细信息

Table 2. Dynamic network data details.

数据集	节点数	边数	时间/d	聚类系数/%
UCI	1899	59835	194	5.68
DNC	2029	39264	982	8.90
Wikipedia	1219241	2284546	4763	0.000837
Enron	384413	1751463	1140	4.96

其中, UCI[38] 是由加利福尼亚大学欧文分校的在线学生社区的用户之间的消息通信而组成的网络. 网络中的节点表示社区用户, 如果用户之间有消息通信, 那么用户之间就会有边连接, 与每条边相关联的时间表示用户之间的通信时间. DNC 是 2016 年民主党全国委员会电子邮件泄漏的电子邮件通信网络. 网络中的节点代表人员, 边代表人员之间的邮件交互. Wikipedia talk, Chinese (Wikipedia)[39] 是中文维基百科的通讯网络, 节点表示中文维基百科的用户, 边表示在某一时刻某一用户在另一用户的对话页上发送了一条消息. Enron [40] 是由 Enron 员工之间发送的电子邮件所组成的电子邮件网络. 和 DNC 一样, 网络中的节点代表员工, 边代表电子邮件. 这些数据集涵盖了多种情况, 例如: UCI 和 DNC 的节点数和边数较少, 而聚类程度较高, 形成较为密集的小网络. 但是它们在持续时间上又有所不同, UCI 的持续时间短, 而 DNC 的持续时间较长. Enron 是节点数和边数较多, 聚类程度也较高的数据集, 形成较为密集的大网络. 而 Wikipedia 是节点数和边数很多, 持续时间很长, 但聚类程度却极低的数据集, 形成稀疏的大网络. 使用这些数据集, 我们可以对模型的鲁棒性进行测试.

根据表 1 中所述的时序网络数据得到 t 时刻的网络拓扑图以及时间信息, 使用平均交互排序 (mean reciprocal rank, MRR) 指标评估链接预测任务的质量. MRR 计算了测试集中真实节点的排名倒数的平均值, 其计算过程如下所示:

$$MRR = \frac{1}{|H|}\sum_{i=1}^{H}\frac{1}{rank_i} \tag{18}$$

其中, H 为测试集中的节点个数, 将目标节点与和其有真实连接的节点之间的余弦相似度进行降序排序, $rank_i$ 则表示了它们的余弦相似度在降序序列中所处的位置. 当测试集中的节点与目标节点间有真实连接时, 其相似度排名应尽可能靠前, 因此 MRR 值越大, 说明链接预测的质量越高, 即网络表示越精准有效. 实验按照时间顺序选取前 80% 的数据作为模型的训练数据, 后 10% 的数据作为验证数据, 其余 10% 的数据作为测试数据. 实验不但与现有的链接预测模型进行了对比, 还与使用了不同信息扩散策略的 DNRLP 模型的变体进行了比较. 并且, 为了验证 DNRLP 模型的准确性, 我们还选取了不同数量的训练数据来与对比模型进行对比. 对于测试集中的每个链接节点对, 我们固定链接一端的节点, 将其看作目标节点, 计算网络中其余节点与该目标节点的余弦相似度, 并进行降序排列.

本文还使用 $Recall@k$ 指标来计算在测试数据集中真实链接占预测结果集中 Top-k 的百分比, 其计算过程如下所示:

$$Recall@k = \frac{1}{|H|}\sum_{i}^{|H|}\sigma\{rank_i \leqslant k\} \tag{19}$$

其中 $\sigma\{rank_i \leqslant k\} = 1$ 表示在预测结果集中真实链接节点的排名 $rank_i$ 小于设定阈值 k. $Recall@k$ 的值越大, 说明链接预测任务的效果越好.

此外, 本文还使用 $Precision@k$ 指标来计算在测试数据集中预测结果占真实链接集中 Top-k 的百分比, 其计算过程如下所示:

$$Precision@k = \frac{1}{|T|}\sum_{i}^{|T|}\sigma\{rank_i \leqslant k\} \tag{20}$$

其中, $\sigma\{rank_i \leqslant k\} = 1$ 表示在预测结果集中真实链接节点的排名 $rank_i$ 小于设定阈值 k. $Precision@k$ 的值越大, 说明链接预测任务的效果越好.

鉴于机器学习模型在链接预测任务中的优异表现, 以及网络表示在常见的网络分析任务中展现出的优异能力, 本文分别使用基于机器学习的链接预测方法和基于网络表示的链接预测方法作为对比方法. 在基于机器学习的方法中, 我们选择两个

经典的机器学习模型, 支持向量机 (SVM) 模型和逻辑回归 (LR) 模型. 在链接预测任务中, 将节点的特征向量作为 SVM 和 LR 模型的输入, 通过节点的特征向量得到节点对的特征向量, 将节点对的特征向量分为有链接和无链接两类, 从而将链接预测问题转变为机器学习中的二分类问题. 在基于网络表示的方法中, 主要通过计算节点表示之间的相似性来进行链接预测, 因此得到更为合适的节点表示是该类方法的主要目的. 为此我们分别选取了具有代表性的三个静态网络表示学习方法和三个动态网络表示学习方法来进行对比, 静态网络表示学习方法包括 node2vec、GCN 和 GraphSAGE, 动态网络表示学习方法包括 DynGEM、GCN-GAN 和 DDNE. Node2vec 是一种优异的图表示学习方法, 它利用随机游走来捕获网络中的邻近性, 并将所有节点映射到一个保持邻近性的低维表示空间中. GCN 构建了一个半监督的节点嵌入模型, 通过对网络拓扑结构和网络节点特征进行编码, 从而得到了含有丰富信息的节点表示. GraphSAGE 通过训练聚合函数将 GCN 扩展到归纳学习任务中, 使其可以直接泛化到未知节点上去. DynGEM 是一种针对时间间隔固定的动态网络的表示学习模型, 它学习到了含有时间信息的节点表示. GCN-GAN 将 GCN、LSTM 和 GAN 相结合, 用 GCN 和来捕获空间结构信息, 用 LSTM 来挖掘时间信息, 最后通过对抗的方式在动态网络中进行链接预测. DDNE 用 GRU 作为编码器来捕获动态网络中的时间信息, 从而在动态网络中进行链接预测. 在上述模型中, SVM、LR、node2vec、GCN 和 GraphSAGE 是适用于静态网络的模型, 因此需要将动态网络转化为静态网络进行实验, 即将所有时刻的网络信息拼接到一个网络中. 而 DynGEM、GCN-GAN、DDNE 以及我们提出的 DNRLP 都是适用于动态网络的模型, 但 DynGEM、GCN-GAN 和 DDNE 中的新链接建立的时间间隔是固定的, 因此在实验中我们忽略动态网络中不同大小的时间间隔. 实验使用网络的邻接矩阵作为模型的输入特征, 将邻接矩阵的行向量作为节点的特征向量. 本文中所有模型的统一实验环境如表 3 所示.

表 3 实验环境设置信息

Table 3. Experimental environment setup information.

项目	设置	数量
操作系统	Ubuntu 16.04	1
CPU	Intel®i7-5280K, 6 核, 12线程	1
硬盘	512GB PLEXTOR®PX-512M6Pro SSD	1
内存	Kingston®8GB DDR4 2400	8
重要程序包	Python 3.7	1
深度学习框架	PyTorch	1

实验中各模型的参数设置如下:

SVM, LR: 根据节点的特征向量得到节点对的特征向量, 将节点对的特征向量分为两类: 有链接的标为 0, 无链接的标为 1. SVM 模型的核函数选用 sigmoid 函数, LR 模型则使用 sag 优化算法来进行求解, 迭代次数设定为 100.

node2vec: 将模型中随机游走的数量定为 20, 随机游走的步长定为 40, 语言模型 Skip-Gram 的窗口大小设定为 10, 最终输出的网络表示维度为 128.

GCN: 将模型中的图卷积网络层数设定为 2, 训练过程迭代次数设定为 500, 学习率设定为 0.01, 输出网络表示的维度设定为 128.

GraphSAGE: 将模型中的搜索深度设定为 2, 邻域采样数量设定为 20, 学习率设定为 0.01, 输出网络表示的维度设定为 128.

DynGEM: 将模型中深度编码器的隐藏层层数设定为 2, 隐藏层单元数分别设定为 [256, 128], 输出的网络表示维度设定为 128.

GCN-GAN: 将模型中的图卷积网络层数设定为 2, LSTM 隐藏层层数设定为 2, 学习率设定为 0.01, 输出的网络表示维度设定为 128.

DDNE: 将模型中深度编码器的隐藏层层数设定为 2, 历史窗口的大小设定为 2, 学习率设定为 0.01, 输出的网络表示维度设定为 128.

DNRLP: 将模型中 LSTM 中的隐藏单元数设定为 128, 新信息扩散过程中的随机游走步长设定为 40, 输出的网络表示的维度设定为 128.

4.2 结果分析

实验结果如表 4 所示. 通过观察对比结果可以看出基于网络表示学习的链接预测方法比基于机器学习的链接预测方法更加有效. 这是因为网络表示学习方法可以对网络节点间的关系进行深入挖掘, 从而得到更加丰富的特征信息. 在基于网络表示学习的链接预测方法中, node2vec 在四个数据

表 4 链接预测 MRR 结果对比

Table 4. Link prediction MRR results comparison.

方法	UCI	DNC	Wikipedia	Enron
Logistic Regression	0.005 1	0.020 9	0.003 7	0.005 2
SVM	0.003 2	0.018 2	0.002 1	0.002 9
Node2Vec	0.004 7	0.019 7	0.003 5	0.003 9
GCN	0.015 9	0.048 4	0.010 1	0.017 6
GraphSAGE	0.016 3	0.049 7	0.012 0	0.018 3
DynGEM	0.015 7	0.028 4	0.010 8	0.014 7
GCN-GAN	0.020 1	0.050 4	0.014 9	0.021 5
DDNE	0.014 2	0.026 8	0.009 6	0.011 6
DNRLP	**0.035 1**	**0.053 9**	**0.018 7**	**0.036 3**

集上均表现一般, 主要因为 node2vec 仅通过随机游走来捕获节点的邻域结构, 没有重视直接相连节点间的信息交互. 且其主要适用于静态网络, 忽略了网络中的动态信息. DynGEM、GCN-GAN 和 DDNE 模型是针对动态网络的表示学习模型, 它们引入了网络中的动态信息, 因而预测效果优于 node2vec, 这说明了动态信息在网络演化中的重要性. 但是 DynGEM 和 DDNE 模型的预测效果不如或者与 GCN 和 GraphSAGE 的效果相似, 这是因为它们仅对网络拓扑图的邻接矩阵进行学习, 只得到了网络的全局拓扑结构信息, 而忽略了网络中的局部信息, 因而学习到的网络特征并没有 GCN 和GraphSAGE 丰富. 而GCN 和GraphSAGE 通过聚合邻居节点的信息来模拟信息在节点间的扩散过程, 既学习到了网络中全局信息也学习到了局部信息, 这表明了局部特征在网络中的重要性, 同时也体现出 GCN 和 GraphSAGE 模型适用于聚类系数较高的邻域信息丰富的网络. 但是 GCN 和 GraphSAGE 忽视了信息传播随时间的衰减, 没有对信息进行遗忘, 而 GCN-GAN 既考虑到了网络中的全局特征和局部特征, 又考虑到了网络演化过程中的动态信息, 因而效果优于 GCN 和 GraphSAGE. 但是 GCN-GAN 模型忽视了时间间隔对信息更新的影响, 而 DNRLP 模型通过信息动态更新模块和信息扩散模块不仅学习到了网络的动态信息, 考虑到了节点邻域所受的影响, 同时还考虑了时间间隔对信息更新的影响, 因此, 该模型在链接预测任务中较其他模型有明显优势. 此外, 我们可以看到, 在 Wikipedia 数据集上所有方法的表现均不佳, 这是因为它的聚类系数太低, 持续时间又太长, 给链接预测任务带来了极大的挑战. 同时对比于其他数据集我们可以看出在聚类系数稍高的情况下, 我们的模型效果要远优于其他所有模型.

本文还在四个数据集上对基于表示学习的链接预测方法中效果较好的几个模型计算了其在不同 k 值下的 *Recall@k* 指标, 实验结果如图 6 所示. 本文所提出的 DNRLP 模型在不同 k 值下的链接预测效果均优于对比模型. 同时随着 k 值的不断增大, *Recall@k* 的值也在不断增大. 我们可以看出, DynGEM 的预测效果与 GraphSAGE 的效果相似, 并且在 DCN 数据集中它的表现较差, 表明了学习局部信息的重要性. 而 GraphSAGE 在 DCN 数据集中的表现优异, 表明了 GraphSAGE 强大的学习邻域信息的能力, 也表明了 GraphSAGE 适用于聚类系数较高的网络. 在不同 k 值下, GCN-GAN 模型的预测效果基本位列第二, 表明了同时考虑空间信息与时间信息的重要性, 而 GCN-GAN 的预测效果要次于 DNRLP, 表明了时间间隔在网络演化过程中的重要性. 上述实验结果表明, DNRLP 模型可以更好的学习网络中的节点信息, 得到含有全局信息、局部信息以及节点偏好信息的节点表示.

此外, 本文还对上述几个模型计算了其在不同 k 值下的 *Precision@k* 指标, 实验结果如图 7 所示. 我们可以看出, *Precision@k* 指标与 *Recall@k* 指标的实验结果相似. 在 DCN 数据集中, 所有方法的表现都比较好, 且当 k 值较小时, DNRLP 与 GraphSAGE、GCN-GAN 的差别不大, 这是因为 DCN 数据的聚类系数较大, 网络中的局部信息相对重要, 而这三个模型均可以通过聚合邻居节点的信息来更新节点表示, 体现了学习网络中局部信息的重要性. 相反在 Wikipedia 数据集上所有方法的表现均不佳, 这是因为它的聚类系数太低, 持续时间又太长, 对进行准确的链接预测有很大的挑战. 在四个数据集上, 本文所提出的 DNRLP 模型在不同 k 值下的 *Precision@k* 指标均优于对比模型, 并且随着 k 值的不断增大, *Precision@k* 的值也在不断增大, 当 k 值较大时, 所提 DNRLP 模型的优势更为明显. 实验结果表明, 在动态网络中 DNRLP 模型可以更为准确地进行链接预测.

为了验证 DNRLP 模型中用基于连接强度的随机游走算法模拟信息扩散过程的有效性, 我们对模型的三个变体进行了对比实验. DNRLP-

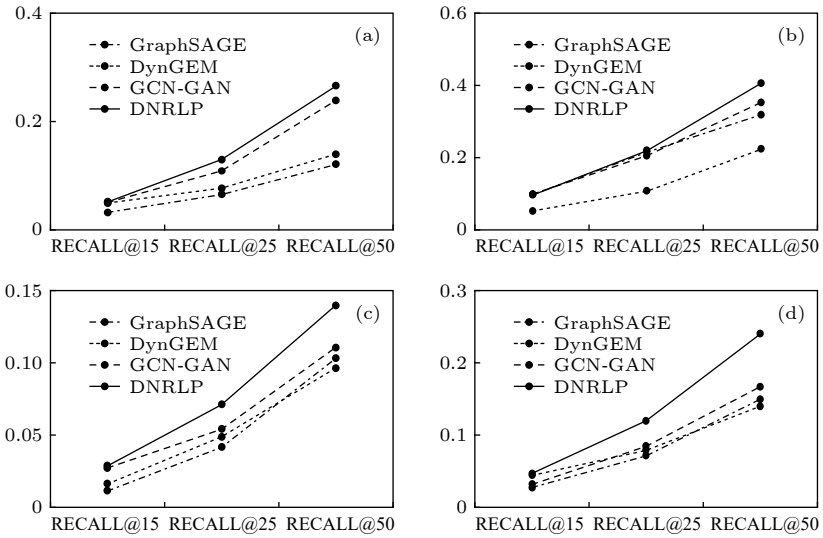

图 6 各数据集上的 *Recall@k* 对比图 (a) UCI 数据集; (b) DNC 数据集; (b) Wikipedia 数据集; (d) Enron 数据集

Fig. 6. *Recall@k* comparison diagram on each data set. (a) UCI dataset; (b) DNC dataset; (b) Wikipedia dataset; (d) Enron dataset.

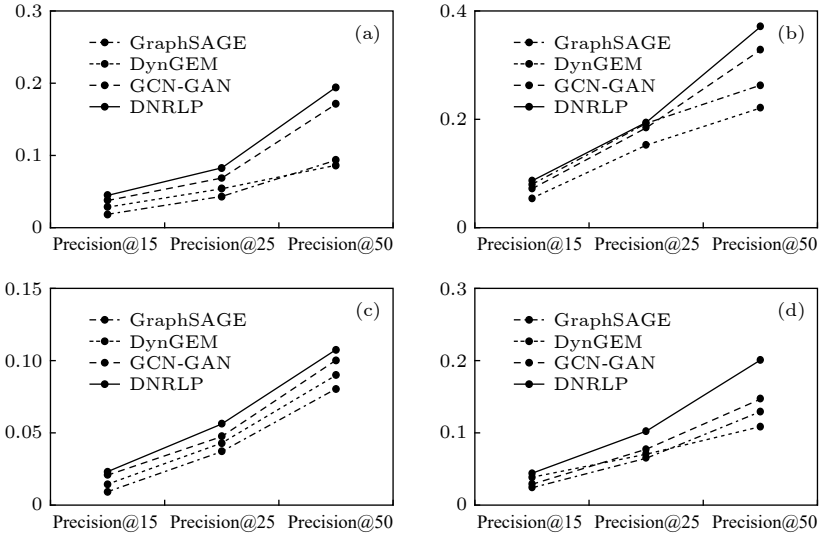

图 7 各数据集上的 *Precision@k* 对比图 (a) UCI 数据集; (b) DNC 数据集; (b) Wikipedia 数据集; (d) Enron 数据集

Fig. 7. *Precision@k* comparison diagram on each data set. (a) UCI dataset; (b) DNC dataset; (b) Wikipedia dataset; (d) Enron dataset.

prop 模型是加入了基于连接强度的随机游走算法的链接预测模型. DNRLP-1 st 模型是使用节点在网络中的一阶邻域模拟信息传播过程的链接预测模型. DNRLP-org 模型为不考虑新信息在网络中的传播的链接预测模型. 对比实验结果如图 8 所示, 可以看出在四个数据集上, DNRLP-prop 模型的预测效果均优于其他两个变体模型, 且 k 值越大, *Recall@k* 的值也越大, 而 DNRLP-org 模型的预测效果最差. DNRLP-org 模型的低预测效果主要是因为它忽略了信息在网络中的扩散过程, 没有将新信息传播到节点邻域中去, 这表明了信息传播在网络中的重要性. DNRLP-prop 模型的预测效果优于 DNRLP-1st 模型的预测效果, 这主要是因为新信息的扩散往往是局部性的, 不仅会对相关节点的一跳邻居产生影响, 也会对与其距离较近的多跳邻居产生影响. 实验结果表明, 动态信息对动态网络的表示学习有着至关重要的作用, 不仅对直接相关的节点有影响, 对其周围一定范围内的节点也有影响. 使用基于连接强度的随机游走算法可以有效地将网络中的动态信息更新到受影响的节点中去.

此外, 为了验证 DNRLP 模型的准确性, 本文

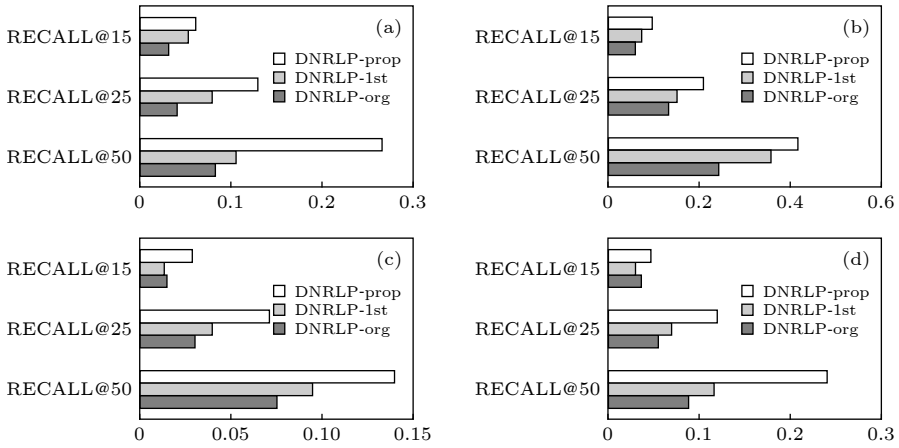

图 8 DNRLP 模型变体的 *Recall@k* 对比图 (a) UCI 数据集; (b) DNC 数据集; (c) Wikipedia 数据集; (d) Enron 数据集

Fig. 8. *Recall@k* comparison diagram of the variants of DNRLP. (a) UCI dataset; (b) DNC dataset; (c) Wikipedia dataset; (d) Enron dataset.

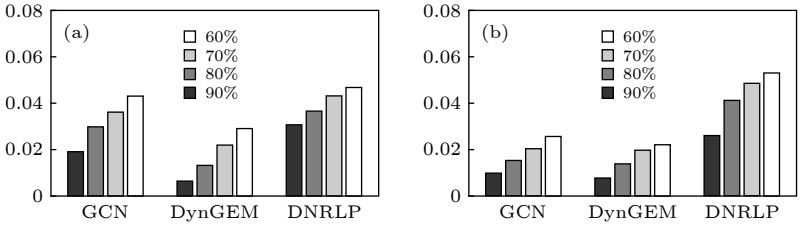

图 9 不同训练率的 MRR 结果对比图 (a) DNC 数据集; (b) Enron 数据集

Fig. 9. MRR results of different training rates. (a) DNC dataset; (b) Enron dataset.

还选取了三个表现较好的模型计算了其在不同比率的训练样本下的 MRR 指标. 我们在两个典型的数据集上, 按照时间顺序分别选取前 60%, 70%, 80%, 90% 的数据作为训练数据, 其余的选择 10% 作为测试数据. 实验结果如图 9 所示, 可以看出在两个数据集上, 随着训练数据比率的增大, MRR 的值也在增大. 并且在任意比率下, DNRLP 的训练效果均优于对比模型, 表现了我们所提模型在链接预测任务中优异的性能.

5 结 论

本文针对现实世界中动态演化的网络提出了一种基于动态网络表示的链接预测模型 DNRLP. 该模型根据动态网络的特性, 在标准 LSTM 单元的基础上引入了基于时间间隔的信息过滤单元, 来决策节点新、旧信息的去留. 此外, DNRLP 模型还考虑了新信息在直接相关节点邻域内的信息传播问题. 本文在四个动态网络公开数据集上对模型的有效性进行了验证, 实验结果表明网络中的全局信息和局部信息对学习良好的网络表示有非常重要的作用, 同时动态网络中的时间信息以及动态信息在网络中的传播对网络节点表示的更新有着极其重要的影响. DNRLP 模型可以学习到动态网络中丰富的信息, 能够有效地对新信息进行快速准确地学习, 在链接预测任务中表现出了明显的优势.

由于现实世界中的网络通常含有多类异构信息, 如社交网络中, 除了含有用户交互产生的网络结构信息以外, 每个用户还具有不同的属性信息, 包括用户的性别、年龄、爱好等. 如何将这些信息加入到链接预测中, 将是一个重要的研究方向.

参考文献

[1] Borgatti S P, Mehra A, Brass D J, Labianca G 2009 *Science* **323** 892
[2] Senator T E 2005 *SIGKDD Explor. Newsl.* **7** 76
[3] Newman M E 2001 *Physical review E* **64** 025102
[4] Adamic L A, Adar E 2003 *Social Networks* **25** 211
[5] Fouss F, Pirotte A, Renders J M, Saerens M 2007 *IEEE Transactions on Knowledge and Data Engineering* **19** 355
[6] Al Hasan M, Zaki M J 2011 *Social Network Data Analytics* (Boston: Springer) p243
[7] Burges C J 1998 *Data Mining and Knowledge Discovery* **2** 121
[8] Freno A, Garriga G, Keller M 2011 *Proceedings of the 25th Neural Information Processing Systems Workshop on Choice*

Models and Preference Learning Granada, Spain, December 12–17, 2011 p1

[9] Hoseini E, Hashemi S, Hamzeh A 2012 *Proceedings of the 26th International Conference on Advanced Information Networking and Applications Workshops* Fukuoka, Japan, March 26–29, 2012 p795

[10] Xu Z, Pu C, Sharafat R R, Li L, Yang J 2017 *Chin. Phys. B* **26** 018902

[11] Lai D R, Shu X, Nardini C 2017 *Chin. Phys. B* **26** 038902

[12] Kovacs I A, Luck K, Spirohn K, Wang Y, Pollis C, Schlabach S, Bian W T, Kim D K, Kishore N, Hao T, Calderwood M A, Vidal M, Barabasi A L 2019 *Nat. Commun.* **10** 1240

[13] Pech R, Hao D, Lee Y L, Yuan Y, Zhou T 2019 *Physica A: Statistical Mechanics and its Applications* **528** 121319

[14] Zhang M H, Chen Y X 2018 *Proceedings of the 32nd Advances in Neural Information Processing Systems* Montreal, Canada, December 2–8, 2018 p5165

[15] Scarselli F, Gori M, Tsoi A C, Hagenbuchner M, Monfardini G 2008 *IEEE Transactions on Neural Networks* **20** 61

[16] Ostapuk N, Yang J, Cudré-Mauroux P 2019 *Proceedings of the 28th The World Wide Web Conference* San Francisco, American, MAY 13–17, 2019 p1398

[17] Gal Y, Islam R, Ghahramani Z 2017 *Proceedings of the 34th International Conference on Machine Learning* Sydney, Australia, August 6–11, 2017 p1183

[18] Gal Y, Ghahramani Z 2016 *Proceedings of the 33rd International Conference on Machine Learning* New York, American, June 19-24, 2016 p1050

[19] Finn C, Abbeel P, Levine S 2017 *Proceedings of the 34th International Conference on Machine Learning* Sydney, Australia, August 6–11, 2017 p1126

[20] Cui P, Wang X, Pei J, Zhu W W 2018 *IEEE Transactions on Knowledge and Data Engineering* **31** 833

[21] Perozzi B, Al-Rfou R, Skiena S 2014 *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* New York, American, August 24–27, 2014 p701

[22] Tang J, Qu M, Wang M Z, Zhang M, Yan J, Mei Q Z 2015 *Proceedings of the 24th international conference on world wide web* Florence, Italy, May 18–22, 2015 p1067

[23] Grover A, Leskovec J 2016 *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* San Francisco, American, August 13–17, 2016 p855

[24] Wang D X, Cui P, Zhu W W 2016 *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* San Francisco, American, August 13–17, 2016 p1225

[25] Kipf T N, Welling M 2016 arXiv: 1609.02907 [cs.LG]

[26] Will H, Ying Z T, Jure L 2017 *Proceedings of the 31st Conference and Workshop on Neural Information Processing Systems* Long Beach, American, December 4–10, 2017 p1024

[27] Schaub M T, Delvenne J-C, Lambiotte R, Barahona M 2019 *Physical Review E* **99** 062308

[28] Srijan Kumar, Zhang X K , Jure Leskovec 2018 arXiv: 1812.02289 [cs.SI]

[29] LI Z Y, Liang X, Xu Z M, Qi J S, Chen Y F 2017 *Chinese Journal of Computers* **40** 805 (in Chinese) [李志宇, 梁循, 徐志明, 齐金山, 陈燕方 2017 计算机学报 **40** 805]

[30] Goyal P, Kamra N, He X R, Liu Y 2018 arXiv: 1805.11273 [cs.SI]

[31] Chen J Y, Zhang J, Xu X H, Fu C B, Zhang D, Zhang Q P, Xuan Q 2019 *IEEE T SYST MAN CY-S1* **49** 1

[32] Hochreiter S, Schmidhuber J 1997 *Neural computation* **9** 1735

[33] Li T S, Zhang J W, Yu P S, Zhang Y, Yan Y H 2018 *IEEE Access* **6** 29219

[34] Dey R, Salemt F M 2017 *Proceedings of the 60th International Midwest Symposium on Circuits and Systems* (*MWSCAS*) Boston, American, August 6–9, 2017 p1597

[35] Lei K, Qin M, Bai B, Zhang G, Yang M 2019 *Proceedings of the IEEE INFOCOM* 2019-*IEEE Conference on Computer Communications* Paris, France, April 29–May 2, 2019 p388

[36] Goodfellow I J, Pouget-Abadie J, Mirza M, Bing X, Warde-Farley D, Ozair S, Courville A, Bengio Y 2014 *Proceedings of the 28th Conference on Neural Information Processing Systems* Montreal, Canada, December 8–13, 2014 p2672

[37] Chang S Y, Zhang Y, Tang J L, Yin D W, Chang Y, Hasegawa-Johnson M A, Huang T S 2017 *Proceedings of the 26th International Conference on World Wide Web* Perth, Australia, April 3–7, 2017 p381

[38] Opsahl T, Panzarasa P 2009 *Social Networks* **31** 155

[39] Sun J, Kunegis J, Staab S 2016 *Proceedings of the 16th International Conference on Data Mining Workshops* Barcelona, Spain, December 12–15, 2016 p128

[40] Klimt B, Yang Y M 2004 *Proceedings of the 15th European Conference on Machine Learning* Pisa, Italy, September 20–24, 2004 p3201

Link prediction model based on dynamic network representation*

Han Zhong-Ming [1)2)†] Li Sheng-Nan [1)] Zheng Chen-Ye [1)]
Duan Da-Gao [1)] Yang Wei-Jie [1)]

1) (*College of Computer and Information Engineering, Beijing Technology and Business University, Beijing* 100048, *China*)

2) (*Beijing Key Laboratory of Food Safety Big Data Technology, Beijing Technology and Business University, Beijing* 100048, *China*)

(Received 29 July 2019; revised manuscript received 7 May 2020)

Abstract

Link prediction is an important issue in network analysis tasks, which aims at detecting missing, spurious or evolving links in a network, based on the topology information of the network and/or the attributes of the nodes. It has been applied to many real-world applications, such as information integration, social network analysis, recommendation systems, and bioinformatics. Existing link prediction methods focus on static networks and ignore the transmission of dynamic information in the network. However, many graphs in practical applications are dynamic and evolve constantly over time. How to capture time information in a dynamic network and improve the accuracy of link prediction remains a conspicuous challenge. To tackle these challenges, we propose a dynamic network representation based link prediction model, named DNRLP. DNRLP can be mainly divided into two modules: a representation learning module on dynamic network and a link prediction module, where the representation learning module is composed of a node information dynamic update unit and a node neighborhood update unit. Node information dynamic update unit leverages the benefits of the long short-term memory (LSTM) in capturing time information and uses a Time Interval based Filter Unit (TIFU) to introduce time interval information between two links, while for the node neighborhood update unit we present a random walk algorithm based on connection strength to simulate the diffusion of dynamic information. Through the above two parts, the model can obtain the node representation at the new moment, then link prediction is performed by the link prediction module by measuring the similarity between the node representations. The experiment uses MRR and *Recall@k* indicators to evaluate performance of model on four public dynamic network datasets. The experiments demonstrate the effectiveness and the credibility of the proposed model in link prediction tasks as compared with the comparison models, the MNR index of the DNRLP is increased by 30.8%. The model proposed in this paper not only learns the dynamic information in the network, but also considers its influence on neighbors and the impact of time interval on information update. Therefore, the model has learned more abundant dynamic information and has obvious advantages for link prediction tasks.

Keywords: link prediction, dynamic network, representation learning, random walk

PACS: 89.75.Hc, 29.85.–c, 89.20.Ff **DOI:** 10.7498/aps.69.20191162

* Project supported by the Natural Science Foundation of Beijing, China (Grant No. 4172016) and the General Project of Beijing Philosophy and Social Science Foundation (Grant No. 14ZHB006).

† Corresponding author. E-mail: hanzhongming@btbu.edu.cn