基于 Word2vec 和粒子群的链路预测算法

贾承丰[1] 韩 华[1] 吕亚楠[1] 张 路[2]

摘 要 链路预测中普遍存在两大问题: 特征提取困难和类别数据不平衡. 本文借鉴文本处理中的深度学习特征提取算法和优化问题中的粒子群算法, 提出一种基于词向量的粒子群优化算法 (Word2vec-PSO). 该方法首先通过随机游走产生网络序列后, 利用 Word2vec 算法对节点序列特征提取. 然后在有监督的条件下, 利用粒子群算法对提取好的特征进行筛选, 并确定重采样的参数来解决类别数据不平衡问题, 并分析了不同链路预测算法的计算复杂性. 最后将本文的算法与基于相似性、基于深度学习、基于不平衡数据的 3 类链路预测算法, 在 4 个不同的时序网络中进行实证对比研究. 结果表明, 本文提出的链路预测算法预测精度较高, 算法更加稳定且具有普适性.

关键词 链路预测, 特征提取, 不平衡问题, 深度学习, 粒子群优化

引用格式 贾承丰, 韩华, 吕亚楠, 张路. 基于 Word2vec 和粒子群的链路预测算法. 自动化学报, 2020, **46**(8): 1703−1713

DOI 10.16383/j.aas.c180187

Link Prediction Algorithm Based on Word2vec and Particle Swarm

JIA Cheng-Feng[1] HAN Hua[1] LV Ya-Nan[1] ZHANG Lu[2]

Abstract There are two major problems in the link prediction: The difficulty of feature extraction and the imbalance of class data. In this paper, an algorithm based on word vector is proposed by using the deep learning feature extraction algorithm in text processing and the particle swarm optimization algorithm in the optimization problem. The method firstly generates a set of node sequences through random walks, and uses the Word2vec algorithm to extract node sequence features. Then, under the supervised conditions, the particle swarm algorithm was used to filter the extracted features, and the resampling parameters were determined to solve the imbalance problem of category data. It also analyzes the computational complexity of different link prediction algorithms. Finally, the algorithm of this paper is compared with three link prediction algorithms based on similarity, deep learning, and unbalanced data, and empirically studied in four different time series networks. The results show that the link prediction algorithm proposed in this paper has more accurate prediction accuracy and is more stable and universal.

Key words Link prediction, feature extraction, imbalance problems, deep learning, particle swarm optimization

Citation Jia Cheng-Feng, Han Hua, Lv Ya-Nan, Zhang Lu. Link prediction algorithm based on Word2vec and particle swarm. *Acta Automatica Sinica*, 2020, **46**(8): 1703−1713

在大数据时代, 随着大量社交网络和信息网络的出现, 链路预测已经成为数据挖掘研究的一个重要方向. 链路预测是指如何通过已知的网络节点以及网络结构等信息, 预测网络中尚未产生连边的两个节点之间产生连接的可能性[1−2]. 近年来, 基于网络节点相似性的链路预测算法被广泛研究. Liben-Nowell 等[3] 根据网络节点和路径的相似性, 最早提出了基于网络拓扑结构的相似性定义方法. 由此开始, 各种基于节点相似性的指标层出不穷, 国内学者 Zhou 等[4] 将 9 种节点相似性的指标应用于 6 个不同的网络中进行链路预测, 发现不同的网络对于各种相似性指标的敏感性不同. 这是由于基于节点相似性的链路预测算法是一种无监督算法, 在不同的网络数据集中预测精度不稳定[4−5].

由于节点相似性算法稳定性不强、准确率不高, 许多研究学者转换思路, 将链路预测问题看作是一种有监督的二分类问题, 利用机器学习算法进行训练、预测. 要利用机器学习算法, 第一步就是要对网络进行特征提取. Benchettara 等[6] 将节点之间的相似性指标作为网络的特征进行提取, 然后利用贝叶斯算法、决策树算法对这些带有标签的节点对进行分类, 将训练出的模型对 Facebook 社交网络进行预测. 因为提取的特征依旧由相似性指标组成, 特征的维度也是人为选取, 网络深层的结构性特性未被表征, 所以算法还是会存在对于不同的网络敏感性不同的问题, Popescul 等[7] 利用逻辑回归算法研究了论文著作网络中两位作者下一时刻合著

收稿日期 2018-04-02 录用日期 2018-09-06
Manuscript received April 2, 2018; accepted September 6, 2018
中央高校基本科研业务费 (185214003, 2018-zy-137) 资助
Supported by Fundamental Research Funds for the Central Universities (185214003, 2018-zy-137)
本文责任编委 祝峰
Recommended by Associate Editor ZHU Feng
1. 武汉理工大学理学院 武汉 430070 2. 武汉安天信息科技公司 武汉 430070
1. School of Science, Wuhan University of Technology, Wuhan 430070 2. Wuhan Antiy Technology Co., Ltd, Wuhan 430070

一篇文章的可能性, 他利用到的网络特征是作者信息 (也就是节点属性信息) 和网络拓扑信息. 但是在实际的链路问题中, 节点的属性信息是很难获取到的, 即使能得到, 节点的属性信息会有噪声, 甚至是虚假信息. 例如电商平台的用户信息是保密的, 或者是注册者提供的虚假身份信息, 所以加入节点属性在实际操作中很难做到. 在人工智能领域, 如果对象的特征难以直接提取, 一般会采用深度学习的方法进行特征提取. Liu 等[8] 利用受限玻尔兹曼机 (Restricted Boltzmann machine, RBM) 加上深度信念神经网络, 对复杂网络进行特征提取, 然后将特征以 Logistic 回归进行训练. 由于训练 RBM 要利用到对比散度算法, 要进行吉布斯采样, 数据要服从高维高斯分布[9], 但是复杂网络的邻接矩阵是一个稀疏矩阵, 数据分布随机性较大, 这就导致需要多次采样, 无疑增加了训练成本. Kipf 等[10] 参照图像处理中的卷积神经网络的思想, 提出类卷积层, 对复杂网络特征提取. 但正因为算法来源于图像, 卷积神经网络注重于图像的边缘[11], 在矩阵中也就是注重相邻元素数值差异大的部分. 差异小的部分可能在池化层就直接合并了, 但是在复杂网络中, 每一条连边的信息都有着自己的作用, 少量连边信息的丢失, 可能会导致部分样本的特征没有被学习到, 从而影响最终分类器的准确率. 另外, 上述算法在突破网络特征提取这一难题之后, 大多数研究方法, 就直接将网络的特征作为分类算法的输入[12−13]. 没有考虑现实网络中, 有连边的节点对数目远远小于无连边的节点对. 传统的机器学习算法会因为类别数目的不平衡这一现象, 而导致决策出现偏差, 详细原因将会在下一节描述. 针对复杂网络特征提取难和类别样本数目不平衡这两大问题. 本文的工作主要如下: 1) 将网络序列化为文本形式, 利用文本处理中的 Word2vec 算法对网络进行特征提取. 2) 为了对上一步的特征进行筛选和处理不平衡问题, 引入粒子群算法对分类器进行优化. 3) 在 4 个时序网络中进行链路预测, 分别对比节点相似性链路预测算法、深度学习提取特征方法、不平衡问题解决方法这三大类算法与本文算法的差异.

1 链路预测问题描述

1.1 问题概述

链路预测可以看作是一个二分类问题: 对于一个无向网络 $G = \langle V, E\rangle$, V代表网络中所有点的集合, 而E则是所有边的集合. 对于所有的节点对, 也就是所有可能产生边的两点集合 $\Omega = V \times V$. 每对节点作为一个样本, 都有自己的标签: 如果 $u \in V$, $v \in V$, 且 $(u, v) \in E$, 则记节点对 (u, v) 的标签为正例, 也就是代表点 u 和点 v 之间有连边; 反之节点对的标签为负例, 也就是代表点 u 和点 v 之间无连边.

本文以时序性网络作为链路预测的对象, 即设 G_i 为 t_i 时刻的网络形态, 集合 $G_{obs} = \langle G_1, \cdots, G_{t-1}\rangle$ 代表前 t 时刻能观察到的网络的总和. 本文将对 G_{obs} 进行特征提取, 作为训练集; 将 G_t 作为验证集调整参数; 将 G_{t+1} 作为测试集, 与在 G_t 的基础上预测出的 $\overline{G}_{t+1}$ 作比较.

1.2 网络特征提取

不同于其他数据挖掘的研究对象, 复杂网络描述的是抽象的点边关系, 网络数据也不像其他数据载体一样具有特定的数据维度 (例如图像有像素矩阵、文本处理中有词典, 其他数据分析问题中有相关属性). 所以对于复杂网络的特征提取是一个困难的工作. 在现有的链路预测研究中, 有很多网络结构属性都被用来刻画网络的特征. 它们反映了不同的网络结构属性, 例如: 基于共同邻居的拓扑属性、基于节点距离的拓扑属性、基于随机游走的拓扑属性, 作为节点相似性进行链路预测. 但是, 针对不同性质的网络而言, 它们对各种相似性指标表现出的敏感性也不同. 这类相似性指标不能体现网络中节点之间错综复杂的非线性关系.

对于像复杂网络这样难以直接提取特征的对象, 近年来都采用深度学习的一系列算法先进行特征提取, 将提取出的样本特征加上样本的标签放入基本的分类算法中再训练. 本文延续这一思路, 借鉴深度学习在文本处理中的应用, 即 Word2vec 算法[14], 将文本中的每一个词类比复杂网络中的每一个节点进行处理, 将词向量作为节点的特征, 进行后一步的分类处理.

1.3 不平衡数据问题描述

现有的研究学者利用有监督学习算法来进行链路预测, 但是他们都没有考虑到复杂网络的特殊性, 由于现实中的网络绝大多数都是一个稀疏网络, 即在所有的节点对中, 有连边的节点对数量远远小于无连边的节点对数量 $|E| \ll (|V| \times |V| - |E|)$. 在学习算法中, 这就是正例的样本数远远小于反例的样本数. 当分类误差作为学习算法的优化对象时, 该算法会忽略正例的贡献, 从而导致分类的结果虽然有一定的准确率但是这种准确率是不可信的. 例如, 医院里有 100 个人体检, 其中 99 个正常, 1 个患病, 如果不经过任何检查, 预测所有人都为正常, 准确率就达到 99 %. 但是, 这种预测是没有任何意义的. 这就是类别数据不平衡导致分类器忽略了少类别样本的贡献度, 结果不具有可信度.

图 1 反映的就是利用文本处理的方法进行链路

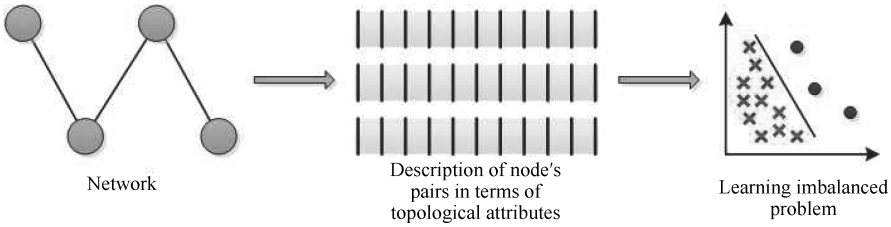


图 1　链路预测算法总流程

Fig. 1　The overall flow of the link prediction algorithm

预测的两大难点问题: 1) 复杂网络难以提取特征; 2) 有监督学习算法难以处理链路预测中的类别不平衡问题.

2　利用 Word2vec 算法提取网络特征

2.1　复杂网络与文本处理的相似性

对于类似复杂网络这种难以直接提取特征的对象, 当今效果最好的算法都是基于深度学习的一系列算法, 例如: 提取图像特征的卷积神经网络、受限玻尔兹曼机; 提取文本特征的方法有基于循环神经网络[15]、LSTM (Long short term memory)、Word2vec 等方法. 本文借鉴深度学习在其他对象的应用, 提出一套对于复杂网络而言完备的特征提取算法. 考虑到复杂网络和文本处理问题有一定的共性, 具体地表现在以下两个方面:

1) 网络数据的最初表现形式 — 邻接矩阵, 它的结构与最初的文本向量化 one-hot 矩阵一样, 都是一个稀疏 0-1 矩阵, 对于每一个维度而言, 文本中的每一个词语相当于网络中的一个节点, 而上述的稀疏矩阵都是体现的点与点或者词与词之间的关系. 对它们进行特征提取, 可以都理解为是对稀疏 0-1 矩阵 ($|V| \times |V|$) 进行降维, 得到一个 $|V| \times d$ 的稠密实数矩阵, 其中, $d \ll |V|$.

2) 对于一篇具有一定规模的文本而言, 只有少量的词语是被反复使用的, 大量的词语是只出现很少的次数. 所以对于一篇长篇幅文本而言, 全文词语的频数是服从幂律分布的. 这点与复杂网络中的节点类似. 往往实际网络都有 BA (Barabási Albert) 无标度网络的特点[16], 即节点的度服从幂律分布. 所以复杂网络的节点与文本处理中的词语在统计学上而言, 具有很强的相似性.

由于复杂网络与文本的相似性, 本文在文本特征提取问题中表现优异的 Word2cev 算法上进行拓展, 从而达到对复杂网络进行特征提取的目的.

2.2　网络点序列的生成与 Huffman 编码

经过分析复杂网络与文本的相似性之后, 需要先对网络数据进行预处理, 使其原本的点边关系具有序列性. 又由于网络节点度与文本词语词频的相似性, 运用 Huffman 编码对网络节点进行二进制编码, 以便于下一步的网络训练.

2.2.1　利用随机游走生成网络点序列

复杂网络从宏观角度看, 描述的是整个复杂系统中所有节点之间的连接关系, 使得直接对整个网络进行特征提取十分困难. 但是如果从微观的角度入手, 整个网络是建立在两个有连接的节点对之上, 进而组成更为复杂的结构. 所以本文从网络局部的链接路径着手, 从微观角度提取更上层的特征.

随机游走作为一种以马尔科夫链为基础的方法, 在复杂网络中的演化机制、网络传播被大量学者所研究. 本文利用 Deepwalk 中随机游走的思想, 产生类似文本的网络序列. 具体而言: 1) 对于网络中的每一个节点 v_i, 从节点 v_i 出发, 以等可能的概率, 游走到邻居节点, 即走到邻居节点 v_j 的概率为 $\frac{1}{k_i}$, 其中 k_i 为点 v_i 的度. 2) 以上一步的终点 v_j 作为起点, 再进行一次随机游走, 可以回到 v_i. 直到路径长度达到 T, 停止随机游走. 这样, 就能产生 $|V|$ 个长度为 T 的序列. 如果在以上的分析中, 将网络中每一个节点都类比为文本中的词语, 那么随机游走产生的序列即是文本中的句子, $|V|$ 个序列的集合就是一个大的文本语料.

2.2.2　节点的 Huffman 编码

在把网络序列化生成文本之后, 就需要对每一个节点进行二进制编码, 才能进入 Word2vec 网络中的最后一层, 具体的算法网络结构会在第 2.3 节中详述. 要区别 $|V|$ 个不同节点, 最简单的二进制编码就是等长编码. 在 Word2vec 算法处理文本的时候, 编码的长度决定了该神经网络的层数, 而神经网络的层数越多, 训练的计算复杂度就越大, 梯度传递消失得越快. 所以要求所有节点的编码总长度尽量短. 但是在网络随机游走产生的序列中, 每个点出现的频率是不相同的, 这是由于网络中每个点的度是不相同的. 度大节点在序列中出现的频率远大于度小节点. 在设计节点编码时, 对于度大的节点用短码, 度小的节点用长码, 从而优化整个网络的编码.

为了使不等长编码为前缀编码 (要求一个节点的编码不能是另一个节点编码的前缀), 可以用网

络中的所有节点作为子叶节点生成一棵编码的二叉树, 为了获得整个网络编码的最短长度, 可以将随机游走序列中的每个节点的出现频率作为树节点的权值赋予在该树节点上, 频数越小权值越小, 权值越小, 叶子就越靠下, 于是频数小编码长, 频数大编码短. 这样, 求整个网络编码的最短长度问题就转化为了求由所有网络节点作为树的叶子节点的 Huffman 树的问题. 利用 Huffman 树设计的二进制前缀编码, 称为 Huffman 编码[17], 它既能满足前缀编码的条件, 又能保证全网络编码总长度最短.

2.3 网络特征的无监督训练

在前面对每个节点进行 Huffman 编码, 它是作为整个 Word2vec 网络的最后的输出层. 然后通过误差修正的训练, 修正输出层之前的所有参数, 输出层之前的网络结构借鉴了 Word2vec 中的 CBOW (Continuous bag-words model) 模型, 目的是为了得到每个节点的特征向量. 与 Deepwalk[13] 中使用的 skip-gram 算法不同, CBOW 算法能够通过多个邻居节点的特征向量决定本节点的特征, 这使得该节点的特征更能体现网络点序列上下文的关系, 从而达到更好的表征效果.

2.3.1 针对复杂网络的 Word2vec 网络结构

传统的基于深度学习的语言模型的目标函数通常取为如下的对数似然函数:

$$L = \sum_{w \in C} \ln p(w|Context(w)) \tag{1}$$

其中 $Context(w)$ 代表的是词 w 的上下文, $p(w|Context(w))$ 代表的是已知上下文的情况下, 出现词语 w 的概率. 而在本文的链路预测方法中, w 就是复杂网络中的节点, $Context(w)$ 则是在随机游走序列中, 点 w 的上下若干邻接点. 我们的目标是得到 w 的最终特征向量 $v(w)$. 为了实现这一目标, 首先要构造条件概率函数 $p(w|Context(w))$.

图 2 给出了 CBOW 模型的网络结构, 它包括三层: 输入层、投影层和输出层. 下面以样本 $(Context(w), w)$ 为例 (这里假设 $Context(w)$ 由点 w 在随机游走序列中前后各 c 个点构成, 相当于文本中一个词语的上下文), 下面对这三层网络结构作简要说明:

1) 输入层: 包含 $Context(w)$ 中 $2c$ 个点的点特征向量 $v(Context(w)_1)$, $v(Context(w)_2)$, $\cdots, v(Context(w)_{2c}) \in \mathbf{R}^d$. 这里的 d 表示的是点特征向量的长度.

2) 投影层: 将输入层的 $2c$ 个向量作求和累加, 即 $X_w = \sum_{i=1}^{2c} v(Context(w)_i) \in \mathbf{R}^m$.

3) 输出层: 输出层对应一棵二叉树, 它是以复杂网络中所有的点当作叶子节点; 叶子节点的度之和记为非叶子节点, 以每个点在网络随机游走序列出现的次数作为权值构造出来的 Hufman 树. 在这棵 Huffman 树中, 叶子节点共 $|V|$ 个, 分别对应复杂网络中的每个点, 非叶子节点 $|V|-1$ 个.

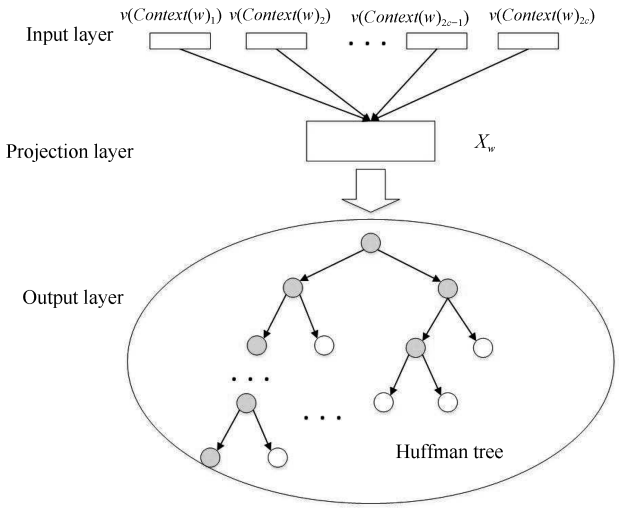


图 2 CBOW 网络结构

Fig. 2 CBOW network structure

2.3.2 梯度计算

在图 2 所示的网络结构已经确定的基础上, 下面就要定义条件概率函数 $p(w|Context(w))$, 更具体地说, 就是要利用向量 $X_w \in \mathbf{R}^m$ 以及 Huffman 树来定义 $p(w|Context(w))$. 对于网络中任意一个点 w, Huffman 树中存在一条从根节点到网络节点对应的路径 p^w, 且路径唯一. 路径 p^w 上存在 $l^w - 1$ 个分支, 其中是点 w 对应的 Huffman 编码的长度, 将每个分支都看作一次二分类, 每一次就产生一个概率, 将这些概率乘起来, 就是所需的 $p(w|Context(w))$.

以二分类的角度考虑 Huffman 树的每一次选择, 那么对于每一个非叶子节点, 就需要为其左右孩子节点指定一个类别, 即哪个是正类 (标签为 1), 哪个是负类 (标签为 0). 这里的 1 或者 0 也可以理解为是 Huffman 编码 d_j^w, 表示的是路径 p^w 中第 j 个节点对应的编码 (根节点不对应编码). 所以以 $sigmoid$ 函数 $\sigma(x)$ 表示一个节点被分为正类的概率是

$$\sigma(X_w^{\mathrm{T}}\theta) = \frac{1}{1+\mathrm{e}^{-X_w^{\mathrm{T}}\theta}} \tag{2}$$

那么 $p(w|Context(w))$ 条件概率即为

$$p(w|Context(w)) = \prod_{j=2}^{l^w} p(d_j^w|X_w, \theta_{j-1}^w) \tag{3}$$

其中

$$p(d_j^w|X_w,\theta_{j-1}^w)=[\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)]^{1-d_j^w}\times [1-\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)]^{d_j^w} \tag{4}$$

代入对数似然函数得到

$$\begin{aligned}L=&\sum_{w\in C}\ln\prod_{j=2}^{l^w}\{[\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)]^{1-d_j^w}[1-\\&\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)]^{d_j^w}\}=\\&\sum_{w\in C}\sum_{j=2}^{l^w}\{(1-d_j^w)\ln[\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w]+\\&d_j^w\ln[1-\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w]\}\end{aligned} \tag{5}$$

为了方便起见, 将花括号中的部分记为 $L(w,j)$, 即

$$\begin{aligned}L(w,j)=&(1-d_j^w)\times\ln[\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)+\\&d_j^w\times\ln[1-\sigma(X_w^{\mathrm{T}}\theta_{j-1}^w)]\end{aligned} \tag{6}$$

到这里已经得到了对数似然函数, 这就是 CBOW 模型用来网络特征提取的目标函数, 接下来只用对此函数进行最大化. 在传统的 Word2vec 算法里采用的是随机梯度上升法. 这种梯度类的算法的关键是给出相应的梯度计算公式, 因此接下来对此函数进行梯度计算.

$$\begin{aligned}\frac{\partial L(w,j)}{\partial\theta_{j-1}^w}=&\frac{\partial}{\partial\theta_{j-1}^w}\{(1-d_j^w)\times\ln[\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]+\\&d_j^w\times\ln[1-\sigma(x_w^{\mathrm{T}}\theta_j^w)]\}=\\&(1-d_j^w)[1-\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]x_w-\\&d_j^w\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)x_w=\\&\{(1-d_j^w)[1-\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]-\\&d_j^w\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)\}x_w=\\&[1-d_j^w-\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]x_w\end{aligned} \tag{7}$$

于是, 每当一个网络序列样本代入之后, θ_{j-1}^w 的更新公式为

$$\theta_{j-1}^w:=\theta_{j-1}^w+\eta[1-d_j^w-\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]x_w \tag{8}$$

其中 η 表示学习率, 接下来考虑 $L(w,j)$ 关于 x_w 的梯度, 在式中可以发现, $L(w,j)$ 中关于变量 x_w 和 θ_{j-1}^w 是对称的 (即两者可以交换位置), 因此, 相应的梯度也只需要交换位置即可.

$$\frac{\partial L(w,j)}{\partial x_w}=[1-d_j^w-\sigma(x_w^{\mathrm{T}}\theta_{j-1}^w)]\theta_{j-1}^w \tag{9}$$

在这里我们能得出 x_w 的更新公式, 但是 x_w 是各点特征向量的累加, 那么对于每一个节点的特征向量更新公式为

$$v(\tilde{w}):=v(\tilde{w})+\eta\sum_{j=2}^{l^w}\frac{\partial L(w,j)}{\partial x_w},\tilde{w}\in Context(w) \tag{10}$$

在经过所有节点序列都输入后, 得到最终的每个点的 $v(w)$ 作为每个点最终的特征向量, 为后续的分类训练提供属性特征. 对于链路预测而言, 预测的对象是节点对, 所以每一对数据对应一个有无连边的标签, 节点对 w_1 和 w_2 的数据由 $v(w_1)$ 和 $v(w_2)$ 两个向量相加得到. 这样就形成一个 d 维的数据对应一个有无连边的标签 (有连边记为 1, 无连边记为 0). 在传统的自然语言处理中, Google 建立的 Word2vec 词向量词典[14] 的维度 $d=20$. 在本文的工作中, 为了保持工作的一致性, 也设置 $d=20$.

3 利用粒子群算法重构数据及选取特征

在第 2 节中, 已经初步得到了特征与标签的数据结构, 在一般的机器学习分类问题中, 此时已经可以直接进行训练分类器进行分类了. 但是对于复杂网络这一特殊的对象以及深度学习产生的多维数据, 直接分类的效果不尽如人意, 主要原因有两点: 1) 真实网络中有连边的节点对数目远小于无连边节点对, 正负例样本的不平衡会导致传统的分类器不能捕捉到少类数据的信息; 2) 深度学习产生的多维数据是无监督的情况下产生的网络特征, 对于链路预测的有监督训练, 可能会出现维度冗余的现象, 影响分类结果. 所以本文利用基于粒子群算法的数据重构和选取特征来解决以上问题.

3.1 降采样与升采样

在其他机器学习研究中, 对于类别不平衡的数据, 通常使用的方法是降采样或者升采样. 两种方法的思想是通过规约样本减少多类样本数目, 或者通过构造样本来增加少类样本的数目. 但是对于复杂网络这一特殊对象而言, 在没有进行深度学习对网络特征提取之前, 节点对都是高维的、离散的、稀疏的向量. 但是经过第 2 节的 Word2vec 模型后, 网络特征都是低维的、连续的、稠密的向量, 这样才能满足升采样和降采样的要求. 所以利用深度学习对网络提取特征也是为网络数据重构提供了可行性.

在现实的数据中, 在多数类的样本中存在着大量的冗余信息和重复信息. 冗余信息会干扰分类器的决策边界的划定, 从而影响分类的精度. 而大量重复的信息, 会使分类算法过多地将分类焦点集中在重复的样本上, 使得最后的分类算法过拟合, 而且会增加运算量. 为了消除这两种隐患, 降采样的思想就是通过去除多数类样本中数据特征相似的样本, 从

而达到删除多余信息, 达到正负类样本数目平衡的目的.

而升采样与过采样的方法正好相反, 升采样的目的是通过增加少数类的样本数目达到正负类样本的平衡. 传统的升采样的做法是, 随机复制少数类的样本加入少数类中. 但是这样会由于样本重复而导致分类器的决策区间过小, 也会导致分类算法的过拟合. 为了应对此问题 Chawla 等[18] 提出 SMOTE 算法, 该算法的思想是对少数类中数据特征相似的样本, 在高维空间中进行连线, 随机地选取连线上的点作为新的样本加入少数类中. 这样产生的新的点既具有少数类样本的数据特征, 又不会和原有样本重复, 是一种插值的思想.

3.2 粒子群优化算法

通过降采样、升采样的方法可以改变网络中正负例节点对的数据分布, 使得样本的数据分布更加均衡. 样本的分布对于后续分类器的训练起着至关重要的作用. 但如果直接使用降采样算法和 SMOTE 算法会出现一些问题: 1) 降采样升采样的比例参数只能通过人工确定, 具有主观性. 2) 采样的算法只解决了数据样本的空间分布, 对于 Word2vec 产生的网络特征并没有很好的选择, 但是特征选择对于不平衡数据的研究也起着至关重要的作用. 所以要达到一个最优预测结果必须进行参数选择, 这就转化为了一个多参数优化问题, 本文选择的粒子群优化算法.

粒子群优化算法 (Particle swarm optimization, PSO) 是一种进化算法, 它通过追随当前搜索到的最优值来寻找全局最优[19]. PSO 初始化为一群随机粒子 (随机解) x_i, 并具有速度 v_i. 然后通过迭代找到最优解. 在每一次的迭代中, 粒子通过跟踪两个 "极值" $(pbest, gbest)$ 来更新自己. 在找到这两个最优值后, 粒子通过下面的公式来更新自己 t 时刻的速度和位置. 更新公式如下:

$$\begin{cases} V_i^{t+1} = w \times v_i^t + c_1 \times r_1 \times \\ \qquad (pbest_i^t - x_i^t) + c_2 \times \\ \qquad r_2 \times (gbest^t - x_i^t) \\ x_i^{t+1} = x_i^t + v_i^{t+1} \end{cases} \tag{11}$$

其中 $i = \{1, 2, \cdots, N\}$, N 代表的是粒子的个数, w 是惯性因子, 也是上一时刻粒子的速度权重, 权衡着全局最优和局部最优的关系. c_1、c_2 分别代表着局部最优的搜索权重和全局最优的搜索权重, r_1、r_2 为产生的 0 到 1 的随机数.

在本文链路预测问题中, 粒子群的解的形式如图 3 所示, 主要有升采样、降采样的比例 $OsLevel$ 和 $UsLevel$, 还有特征向量的选择 $(f_1, f_2, \cdots, f_n)$, 当选择第 i 个特征的时候, $f_1 = 1$, 否则为 0. 但是在 PSO 算法中, 要求所有的变量都为连续的变量, 像 f_i 这种离散的二值变量要转化为连续变量, 本文使用的是 sigmoid 函数进行转化:

$$v_i^{t'} = sig(v_i^t) = \frac{1}{1 + \mathrm{e}^{-v_i^t}}$$

$$x_i^{t+1} = \begin{cases} 1, & \text{若 } r_i < v_i^{t\prime} \\ 0, & \text{否则} \end{cases} \tag{12}$$

$OsLevel$	$UsLevel$	f_1	f_2	$\cdots$	f_{n-1}	f_n

图 3 PSO 中解的表达形式

Fig. 3 The expression of the solution in PSO

本文粒子群的优化目标无疑是使得链路预测算法预测得最精准, 链路预测的结果评判指标为 AUC (Area under the curve) 值[20], 它能最好地反应类别不平衡问题中的预测结果, 而不是单单只关注多少样本被分类正确, 在这里我们也是求 AUC 的最大值作为最后的优化目标. 所以我们将每一时刻 t 的重构样本与选取的特征放入分类器中, 求得每一时刻的最优参数, 并计算 AUC 值, 此处用的基础分类算法是 C4.5 决策树算法.

在粒子群优化中, 利用 AUC 作为优化目标函数, 来进行优化. 为了确定参数以及防止过拟合, 在优化过程中验证分类的结果的时候, 把训练集分为训练子集 Trt (70 %) 和验证子集 Trv (30 %). 对训练子集 Trt 进行采样和特征选择, 构造新的数据集 BalTrt, 并训练分类器, 之后再验证子集 Trv 进行测试得到分类结果, 和重新确定各个参数, 具体算法描述如下:

输入. Word2vec 提取点特征, 寻优次数 T = 3 000, 采样次数 $Numk$ = 30, 粒子的个数 SN = 25, metric M(AUC), 采样步长 step.

输出. 最终的分类器 C.

1) 划分数据集 Dataset 成训练集 TrDataset 和测试集 TeDataset;

2) 初始化每个解 x_i $(i = 1, 2, \cdots, SN)$. 每个解的维度 $= d + 2 = 22$.

3) 全局最优解 BestM = 0;

for $i = 1$ to T

for $j = 1$ to SN

4) 获得当前解 x_j 中的升采样降采样比例, 也就是 $x_{j,1}$、$x_{j,2}$.

for $k = 1$ to $Numk$

根据升采样、降采样的比例以及特征对 Trt 采样和特征选择操作, 生成新的训练子集 BalTrt.

决策树构建分类器, 在测试集中得到分类结果 M_0.

end for

5) $M = (M + M_0)/k$, 即把每次交叉验证的 M 的平均值作为对应当前解 x_j 在第 i 轮的适应度值.

end for

6) 根据式 (11) 以及 step 更新粒子群中每个解的位置, 及更新全局最优解 BestM.

end for

7) 得到 BestM 对应的最优解, 包括升采样降采样比例 ($OsLevel$, $UsLevel$) 以及特征选择的策略 $(f_1, f_2, \cdots, f_n)$. 构建最终的分类器 C.

在步骤介绍完成后, 下面对该粒子群特征提取的方法进行时间复杂度分析. 在上述粒子群计算过程中, 一个粒子每维分量需要进行 5 次乘法运算, 5 次加法运算, 设计算机一次乘法运算时间为 T_m, 一次加法时间为 T_n, 决策树算法的时间复杂度为 $d \times Numk$ 所以本算法完成优化的平均时间为 $d^2 \times T \times SN \times Numk \times (5T_m + 5T_n)$.

4　实验设计与结果分析

有了上述的基于 Word2vec 和粒子群 (Word2vec-PSO) 的链路预测算法之后, 我们以 G_{obs} 中的节点对作为训练集, 利用 G_t 网络中所有节点对的属性作为测试集的特征, 通过我们的链路预测算法得到了预测后的节点对连边概率 R_1 和非连边概率 R_2. 并与 G_{t+1} 网络的真实连接情况共同分析, 评价算法的优劣. 另外对于同一个数据集, 我们利用了基于节点相似性的链路预测算法和其他几种专门针对链路预测的数据不平衡问题设计的其他算法, 得到的实验结果与本文的进行比较分析.

首先, 利用在时序网络中被广泛研究的 4 个真实网络建立好训练集和测试集; 再利用已被提出的链路预测学习算法与本文提出的新的集成分类算法进行链路预测, 计算各类指标; 最后通过对比研究它们各自在传统的链路预测指标上的表现.

4.1　实验数据

本文选取了 4 个不同规模的大型真实网络, 其中包括 3 个著作权网络 (Citeseer[21]、Cora[22]、Pubmed[23]) 和一个虚拟社交网络 (2017 年 9 月 1 日～2018 年 2 月 1 日部分新浪微博持续活跃用户关注网络). 在著作权网络中节点代表每一位论文作者, 边代表作者之间合著过至少一篇文章 (不考虑多次合作也就是不考虑权重的影响). 虚拟社交网络中每一个节点是一位微博用户, 边代表两用户之间至少有一方关注, 该数据利用 Python 中的 Beautiful Soup 包爬取. 它们的网络基本特征见表 1.

4.2　相关链路预测算法及评判指标

由于本文主要解决的是链路预测中的两大问题: 1) 难以对复杂网络提取特征; 2) 网络节点对有无连边的数目相差太大. 所以要与本文对比的其他链路预测算法也是围绕这两大问题进行. 不失一般性, 我们也将基于节点相似性的链路预测算法纳入比较范围. 对于 3 类不同的链路预测算法的概况分别见表 2～4.

在第 3 节, 已经描述了 AUC 指标对于不平衡数据分类的重要性. 但是 AUC 指标更侧重于全局样本的正确分类情况, 在不同的链路预测问题中, 还有其他的要求. 例如在微博推荐系统中, 要求算法能最精准地推荐 “可能认识的人”, 即有连边的节点对, 对于 “可能不认识的人” 却不关心. 对于这种要求, Precision 指标也应该列入分类结果分析中. 它关注的是前 L 个预测边中预测准确的比例, 后文统一用 P 代替.

表 1　本文用到的数据集概况

Table 1　Overview of the data sets used in this article

数据集	网络类型	节点数	边数
Citeseer	著作网络	3 327	4 732
Cora	著作网络	2 708	5 429
Pubmed	著作网络	19 717	44 338
微博关系网络	虚拟社交网络	65 775	266 144

表 2　相似性预测链路算法

Table 2　Similarity link prediction algorithm

链路预测算法	算法公式	算法概述
Common neighbor[3]	$S_{ij} = \lvert\Gamma(i) \cap \Gamma(j)\rvert$	共同邻居节点的个数
Jaccard's coefficient[24]	$S_{ij} = \frac{\lvert\Gamma(i) \cap \Gamma(j)\rvert}{\lvert\Gamma(i) \cup \Gamma(j)\rvert}$	邻居节点集合的交集与邻居节点集合的并集的比值
Adamic Adar[25]	$S_{ij} = \sum_{x \in \Gamma(i) \cup \Gamma(j)} \frac{1}{\ln k_x}$	共同邻居节点度的对数的倒数之和

表 3 复杂网络其他特征提取算法

Table 3 Other feature extraction algorithms for complex networks

链路预测算法	算法公式	算法概述
Likelihood supervised machine learning[7]	Similarity index+ Classical machine learning	以相似性指标为特征, 利用传统机器学习算法进行训练
RBM-DBN link prediction[8]	$F(v) = -\sum_i v_i a_i - \sum_j \ln(1 + E^{h_j})$	利用受限玻尔兹曼机提取特征, 深度信念网络进行训练
Convolutional networks[9]	$Z = f(X, A) = softmax(\hat{A}ReLu(\hat{A}XW^{(0)}))W$	利用卷积神经网络对网络提取特征, 卷积层后面连接 softmax 层进行分类

表 4 链路预测其他处理类别不平衡问题算法

Table 4 Link prediction other processing category imbalance problem algorithm

链路预测算法	算法公式	算法概述
AUC-logistic regression[26]	$\varphi_{AUC\text{-}Logistic} = \sum_{(i,j,z)\in T} 1(x_i^{\mathrm{T}} M x_j - x_i^{\mathrm{T}} M x_z)$	对特征矩阵 M 进行逻辑回归训练, 目标函数是 AUC 值
Rank-SVM[27]	$\varphi_{SVM} = \sum_{(i,j,z)\in T} \max(0, 1 + x_i^{\mathrm{T}} M x_z - x_i^{\mathrm{T}} M x_j)$	利用支持向量机学习节点连接概率的大小排序
Entropy algorithm[28]	$\min_M L(M) = \lambda\Omega(M) + \sum_{q\in V} \varphi(S^R(M), R^q)$	利用交叉熵算法, 使得网络全局交叉熵损失最小化, 处理类别不平衡节点对

从算法的计算复杂性分析机器学习、深度学习链路预测算法的时间复杂度, 是远大于相似性链路预测算法的. 这也是相似性链路预测算法的优点, 但是从预测精度的角度来说, 相似性链路预测算法不能对未来的连边做到有效的判断, 下面的实验结果也会支撑这一结果. 所以本文着重比较机器学习相关的链路预测算法的时间复杂度.

对于 Likelihood supervised machine learning 算法, 时间复杂度为 $O(n^2)$, 其中 n 为提取的相似性指标的个数. 对于 RBM-DBN link prediction 算法, 时间复杂度为 $O(n^d \times m)$, 其中 n 为特征维度, d 为受限玻尔兹曼机的深度, m 为深度信念网络神经元个数. Convolutional networks 算法, 时间复杂度为 $O(\sum_{l=1}^{D} M_l^2 \times K_l^2 \times C_{l-1} \times C_l)$, 其中 D 为网络的总层数, l 为网络的第 l 层. M 为特征图边长, K 为卷积核维度, C 为每层输出通道的个数. 由前文可知, 本文的 Word2vec 方法提取特征, 由于使用了 Huffman 二叉树, 大大减小了 softmax 层单个计算时间, 加上后面的粒子群优化, 总时间复杂度为 $O(\ln_2 |V| + d^2 \times T \times SN \times Numk)$. 从算法的时间复杂度上分析, 其实本文的算法在深度学习算法中, 属于计算复杂度较高的算法, 运行时间并不占优. 但是相较于受限玻尔兹曼机和卷积神经网络的参数较为固定, 文本的参数可调范围更大, 可以做到运行时间和预测精度的平衡.

4.3 实验结果

首先, 我们先要用 Word2vec 算法对网络提取特征, 理论上来说, 只要最后的链路预测算法的预测结果表现好, 就能说明特征提取的工作优秀, 因为特征工程是所有机器学习算法的先决条件. 但是为了算法思想的连贯性, 本文给出直观的特征提取的可视化实例, 对实验结果进行补充.

图 4 展示的是利用 Word2vec 算法对 Karate 网络进行特征提取 (特征为 2 维) 的结果, 可以看到同一社团中的节点, 在二维特征空间中, 空间距离也很近. 这说明相似的节点提取出了相似的网络特征, 特征工程比较成功. 而且在实际链路预测时, 特征维度会远远大于 2, 随着特征维度的增加, 节点的属性将会描述得更加详细.

接下来, 利用本文的 Word2vec-PSO 算法对实际的 4 个大型网络进行链路预测, 比较不同算法的优劣. 结果见表 5.

加粗部分的数值为预测效果最好的算法. 可以看出不管是对于 AUC 指标, 或者是 Precision 指标, 本文提出的 Word2vec-PSO 链路预测算法基本上都有精准且稳定的表现. 只对于 Citeseer 著作网络来说, Rank-SVM 的表现更加突出. 产生这种情况的原因可能有两种:

1) 从算法理论的角度来说: Rank-SVM 算法相较于本文提出的算法, 更加注重最有可能连边的节

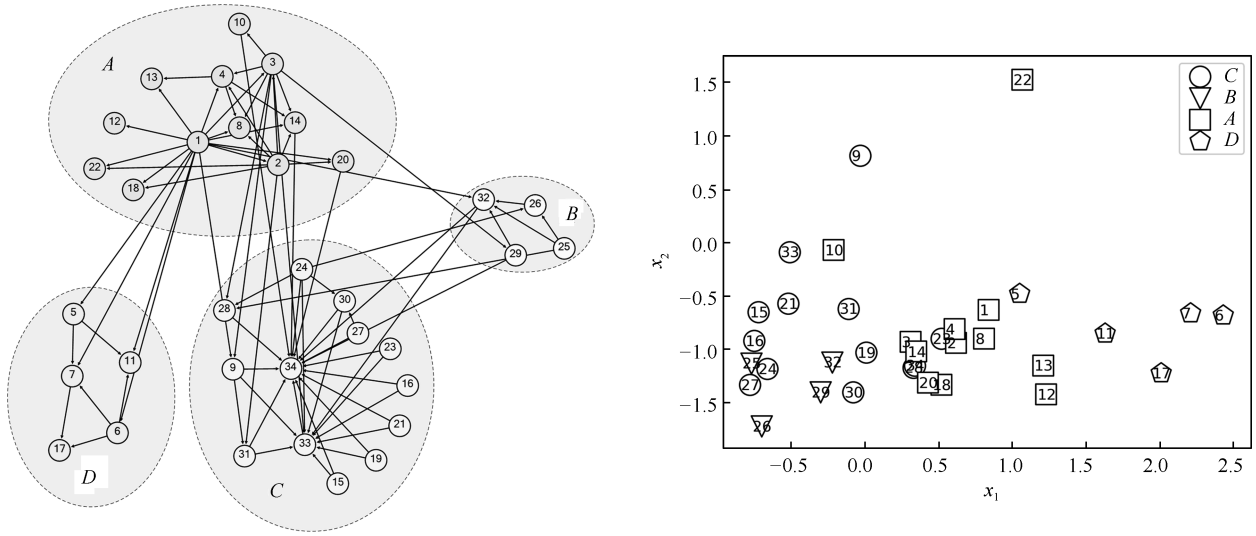


图 4　Karate 网络特征提取

Fig. 4　Karate network feature extraction

表 5　本算法与其他链路预测算法结果对比

Table 5　Comparison of the algorithm with other link prediction algorithms

链路预测算法	Citeseer		Cora		Pubmed		Weibo	
	AUC	P	AUC	P	AUC	P	AUC	P
Common neighbor	0.567	0.632	0.616	0.696	0.561	0.669	0.542	0.615
Jaccard′s	0.568	0.651	0.616	0.694	0.564	0.668	0.540	0.618
Adamic Adar	0.675	0.690	0.679	0.711	0.584	0.702	0.559	0.621
Common neighbor	0.656	0.759	0.715	0.784	0.827	0.853	0.687	0.741
Jaccard′s	0.673	0.753	0.524	0.612	0.643	0.781	0.664	0.727
Adamic Adar	0.789	0.731	0.744	0.829	0.687	0.778	0.765	0.854
Common neighbor	0.752	0.857	0.711	0.773	0.781	0.829	0.545	0.628
Jaccard′s	0.861	**0.934**	0.689	0.746	0.718	0.805	0.712	0.802
Adamic Adar	0.762	0.860	0.797	0.805	0.810	0.865	0.786	0.851
Word2vec-PSO	**0.818**	0.872	**0.801**	**0.823**	**0.867**	**0.913**	**0.833**	**0.875**

点对的属性, 也就是说它利用的支持向量机的支持向量对正例特征最明显向量敏感, 所以分类时能保证特征最 “突出” 的正例能被有效识别, 也正好符合 Precision 指标的定义; 而本文的 Word2vec-PSO 算法, 由于算法的优化目标就是最终的 AUC, 也就是说更加注重全局的预测表现, 因为决策边界试图把正负例特征模糊的样本也能区分开来, 对于样本的学习会注重每一个个体, 存在着过拟合的隐患, 这也是本算法不可避免的一个理论弊端.

2) 从实验数据的角度来说, Citeseer 网络节点相对比较少, 而且连边也比较稀疏. 对于深度学习提取网络特征而言, 需要大型的网络随机游走产生更丰富的节点序列, 这样才能生成大样本的语料库, 进行角度的特征提取. 而样本容量过少会导致特征提取得不好. 这也是上文对 Karate 网络特征提取效果不是特别准确的原因. 这也揭示了本算法的另一个特点, 对于大型的网络, 基于深度学习提取特征的算法会有更好的表现.

虽然上文分析了本文算法在实际应用中可能出现的问题, 但从实验结果看. 即使是在 Citeseer 数据集上的预测表现没有其他的算法好, 但是 0.872 的 Precision 指标本身也算是达到了很好的预测目的. 而反观 Rank-SVM 算法和其他链路预测算法, 即使是它们在某一两个网络中表现良好, 但是在其他网络中表现会很差, 体现出了算法的不稳定和特殊的适用性. 这是因为其他非深度学习算法特征提取时是人工特征提取, 将网络的某一些指标作为节点属性, 当网络对这些指标敏感的时候, 分类时就会有很好的效果. 但是一旦网络对这些人工挑选的属性不敏感的时候, 算法的表现会有很强的波动性. 所

以本文基于深度学习的特征提取方法, 对于不同性质、不同敏感度的网络都适用, 且算法表现稳定, 说明了本文提出的 Word2vec-PSO 链路预测算法具有一定的普适性.

实验最后, 针对不同算法对于网络特征提取的时间, 我们也做了相关实验进行比较. 由于本文选取的是 4 个不同的网络, 4 个节点分布较为离散, 而且是固定值. 但是特征提取的时间与网络节点的个数息息相关, 单从 4 个网络的特征提取时间来验证算法时间效率的问题有失偏颇. 这里, 本文效仿 Simonya 等[11] 的工作, 生成点个数连续的 ER 网络. 网络点的个数从 100 到 100 万个逐步增加, 记录特征提取完成时间, 每个网络重复 10 次提取特征, 对耗费的时间求均值, 通过时间曲线对比研究各算法的运行效率.

图 5 展现的是 4 种链路预测特征提取算法运行时间的对比图, 横坐标为产生的 ER 网络节点的个数, 处理为以 10 为底的对数, 纵坐标为时间, 也处理为以 10 为底的对数. 用来计算的机器配置为 CentOS7.4 系统, 2 核 CPU, 4 GB 内存, 2 GB 英伟达 Tesla GPU.

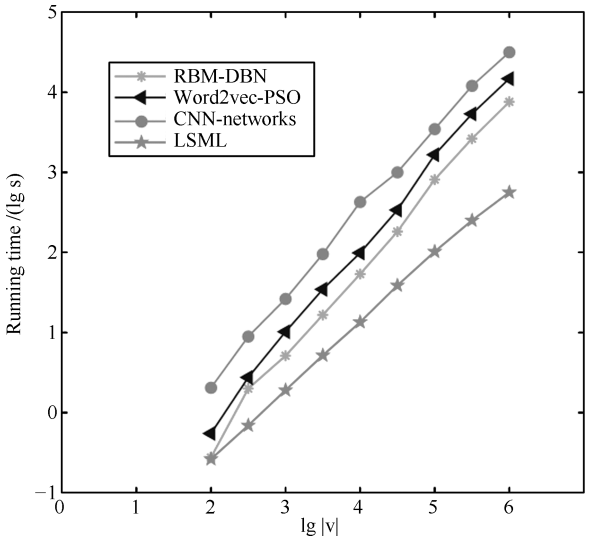


图 5 四种链路预测特征提取算法时间对比

Fig. 5 Time comparison of four link prediction feature extraction algorithms

可以看出, LSML 由于直接利用各个相似性指标, 计算较为简单, 用时较深度学习提取特征系列算法更短, 而且运行效率差异明显. 而本文提出的 Word2vec 特征提取算法, 计算效率在 RBM-DBM 和 CNN-networks 算法之间. 在计算复杂性上表现一般, 称不上是快速的算法. 不过, 虽然三种算法在计算效率上可以说是不相伯仲, 但是在前文分析的准确性上, 本文提出的算法还是在预测精度上更胜一筹. 相比较相似性特征提取的方法, 更是遥遥领先.

5 结论

本文针对复杂网络特征提取难和类别样本数目不平衡这两大问题. 主要做了以下工作并得到一些结论, 具体为: 1) 将网络序列化为文本形式, 利用文本处理中的 Word2vec 算法对网络进行特征提取, 将抽象的点边关系的具体化、数值化. 2) 利用粒子群算法对提取出的特征进行筛选, 并对原始数据进行重构, 解决了链路预测中的样本类别不平衡问题. 3) 在 4 个时序网络中进行链路预测, 分别对比节点相似性链路预测算法、深度学习提取特征方法、不平衡问题解决方法这 3 大类算法与本文算法的差异. 3 大类算法各有各的优势, 因为本文同时考虑到了特征提取和类别不平衡两个链路预测主要问题, 从综合角度来说, 算法理论更全面, 从大型网络的链路预测实验中, 也体现了本算法良好的预测表现.

本文借鉴的是文本处理中的深度学习算法, 将复杂网络对象类比为文本进行研究, 可以看出文本与网络的共性. 从另一角度考虑, 一部分研究学者也将文本转化为了复杂网络进行研究. 本算法还可以应用到文本网络中, 对文本进行链路预测, 从而实现文本语义识别、词性标注等一些自然语言处理问题.

References

1 Fayyad U. *ACM SIGKDD Explorations Newsletter*. New York: ACM, 2002

2 Albert R, Barabási A L. Statistical mechanics of complex networks. *Review of Modern Physics*, 2002, **74**(1): 47

3 Liben-Nowell D, Kleinberg J. The link prediction problem for social networks. In: Proceedings of the 12th International Conference on Information and Knowledge Management. New Orleans, LA, USA: ACM, 2003. 556−559

4 Zhou T, Lv L Y, Zhang Y C. Predicting missing links via local information. *The European Physical Journal B*, 2009, **71**(4): 623−630

5 Keck F, Bouchez A, Franc A, Rimet F. Linking phylogenetic similarity and pollution sensitivity to develop ecological assessment methods: A test with river diatoms. *Journal of Applied Ecology*, 2016, **53**(3): 856−864

6 Benchettara N, Kanawati R, Rouveirol C. Supervised machine learning applied to link prediction in bipartite social networks. In: Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining. Odense, Denmark: IEEE, 2010. 326−330

7 Popescul R, Ungar L H. Statistical relational learning for link prediction. In: Proceedings of the 2003 Workshop on Learning Statistical Models from Relational Data. IJCAI, 2003.

8 Liu F, Liu B Q, Sun C J, Liu M, Wang X L. Deep learning approaches for link prediction in social network services. *International Conference on Neural Information Processing*. Berlin Heidelberg: Springer, 2013. 425−432

9 Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006, **18**(7): 1527−1554

10 Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. arXiv: 1609.02907, 2016

11 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv: 1409.1556, 2014

12 Lowrance C J, Lauf A P, Kantardzic M. A fuzzy-based machine learning model for robot prediction of link quality. In: Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence. Athens, Greece: IEEE, 2017. 1−8

13 Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM, 2014: 701−710

14 Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv: 1301.3781, 2013

15 Hochreiter S, and Schmidhuber J. Long short-term memory. *Neural Computation*, **1997**(9): 1735−1780

16 Barabási A L, Albert R, Jeong H. Scale-free characteristics of random networks: The topology of the world-wide web. *Physica A: Statistical Mechanics and Its Applications*, 2000, **281**(1−4): 69−77

17 Vermeulen T, Huffman E H. Ion exchange column performance – hydrogen cycle rates in nonaqueous solvents. *Industrial & Engineering Chemistry*, 1953, **45**(8): 1658−1664

18 Chawla N V, Bowyer K W, Hall L O, Kegelmeyer W P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002, **16**: 321−357

19 Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of the 2002 IEEE International Conference on Neural Networks. Perth, WA, Australia: IEEE, 2002. 1942−1948

20 Golicher D, Ford A, Cayuela L, Newton A. Pseudo-absences, pseudo-models and pseudo-niches: Pitfalls of model selection based on the area under the curve. *International Journal of Geographical Information Science*, 2012, **26**(11): 2049−2063

21 Fiala D. Mining citation information from CiteSeer data. *Scientometrics*, 2011, **86**(3): 553−562

22 Lunin V V, Dobrovetsky E, Khutoreskaya G, Zhang R G, Joachimiak A, Doyle D A, et al. Crystal structure of the CorA Mg^{2+} transporter. *Nature*, 2006, **440**(7085): 833−837

23 Falk E, Shah P K, Fuster V. Coronary plaque disruption. *Circulation*, 1995, **92**(3): 657−671

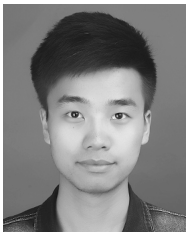
24 Jaccard P. Etude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Societe Vaudoise des Sciences Naturelles*, 1901, **37**(142): 547−579

25 Adamic L A, Adar E. Friends and neighbors on the Web. *Social Networks*, 2003, **25**(3): 211−230

26 Menon A K, Elkan C. Link prediction via matrix factorization. In: Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases. Berlin, Heidelberg: Springer-Verlag, 2011. 437−452

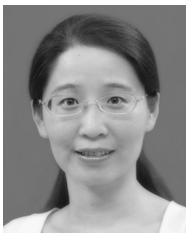
27 Yazdani M, Collobert R, Popescubelis A. Learning to rank on network data. *International Journal of Information Management*, 2013, **6**(3): 187−188

28 Li B P, Chaudhuri S, Tewari A. Handling class imbalance in link prediction using learning to rank techniques. arXiv: 1511.04383, 2016



贾承丰 武汉理工大学理学院硕士研究生. 主要研究方向为复杂网络, 机器学习. E-mail: 13986076510@163.com

(**JIA Cheng-Feng** Master student at the School of Science, Wuhan University of Technology. His research interest covers complex network and machine learning.)

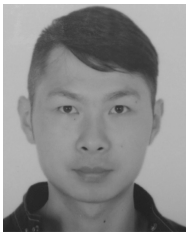


韩 华 博士, 武汉理工大学理学院教授. 主要研究方向为系统预测, 复杂网络, 经济决策. 本文通信作者.
E-mail: hhua@whut.com

(**HAN Hua** Ph. D., professor at the School of Science, Wuhan University of Technology. Her research interest covers system prediction, complex network, economic decision-making. Corresponding author of this paper.)

吕亚楠 武汉理工大学理学院硕士研究生. 主要研究方向为链路预测, 复杂网络. E-mail: lyn@whut.com

(**LV Ya-Nan** Master student at the School of Science, Wuhan University of Technology. Her research interest covers link prediction and complex network.)



张 路 武汉安天科技公司数据挖掘工程师. 主要研究方向为自然语言处理, 机器学习. E-mail: 17838907371@163.com

(**ZHANG Lu** The data mining engineer of Wuhan Antiy Technology Co., Ltd. His research interest covers natural language processing and machine learning.)