文章编号:1006-2467(2020)02-0111-06 DOI: 10.16183/j.cnki.jsjtu.2020.02.001

基于网络最大流的作者同名区分算法

全锦琪， 傅洛伊， 甘小莺， 王新兵
(上海交通大学 电子信息与电气工程学院，上海 200240)

摘 要：为了降低不同学者实体之间的共享特征(如机构、发表会议等)给同名区分带来的影响，提出一种基于网络最大流的同名区分算法.该算法将论文实体及其特征融合成一张网络图，根据特征节点的被共享程度设定不同的容量，再计算论文节点间的最大流量，并基于最大流量进行层次聚类.实验结果表明：该算法在精准率和召回率上有较为均衡的表现，具有较好的综合性能.

关键词：同名区分；最大流；聚类；学术网络
中图分类号：TP 391 **文献标志码**：A

A Network Maximum Flow Based Approach for Author Name Disambiguation

QUAN Jinqi， *FU Luoyi*， *GAN Xiaoying*， *WANG Xinbing*
(School of Electronic, Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: In order to reduce the influence of sharing features (organizations, conferences, etc.) among different author entities on author name disambiguation, an algorithm based on network maximum flow is proposed in this paper. The algorithm puts the paper entities and features into a network graph, and sets the capacity of feature nodes based on the sharing degree. And then, it calculates maximum flow between each paper nodes and does clustering based on maximum flow. The experiment results show that the proposed algorithm has a more balanced performance on accuracy and recall, and has better overall performance.

Key words: name disambiguation; maximum flow; clustering; academic network

随着互联网技术的快速发展，大量学术论文以电子文档的方式保存在各出版社的电子数据库中，如 ACM 电子数据库(DL)、IEEE DL、Springer 数据库等.为了方便研究人员对不同数据库中的论文进行统一检索，很多学术搜索引擎也随之产生，如谷歌学术、微软学术、DBLP、Acemap 等.通过这些学术搜索引擎，研究人员可以快速精准地检索出自己想要的论文，不仅可以通过标题进行精确检索，也可通过关键词进行模糊检索，还可以按照学者姓名检索出该学者发表的所有论文.

然而，由于许多学者可能拥有完全一样的名字，即学者的名字具有歧义性，给学术搜索引擎在数据整合及检索上带来很大的困扰.在学术论文数据整合过程中，若不能很好地对同名学者进行区分的话，那么当用户想检索某一位学者发表的所有论文时，往往得不到精确的结果.另外，目前也有许多研究人

收稿日期:2018-07-24
作者简介:全锦琪(1994-)，男，广东省茂名市人，硕士生，主要研究方向为数据挖掘.
通信作者:王新兵，男，教授，博士生导师，E-mail:xwang8@sjtu.edu.cn.

员从事针对学者的数据挖掘研究，例如学者师承关系挖掘、学者未来影响力预测等. 这些研究都需要以能够精准获得某位学者发表过的所有论文为前提. 因此，对同名学者的区分具有很重要的研究意义.

目前，已有许多国内外的研究人员对学者同名区分问题展开了研究. Yin 等[1]提出了同名区分框架 DISTINCT，该框架结合 Jaccard 系数及随机游走来计算集合的相似度，利用支持向量机(SVM)训练出各特征的权重，再进行层次聚类. Tang 等[2]提出一种基于隐马尔科夫随机场的模型，模型每次迭代分配标签，寻找出最大化后验概率的分配方案. Fan 等[3]提出了基于图论的 GHOST 框架，该框架首先取得两点间 k 跳内的所有简单路径，然后按照规则剔除无效边，再采用仿射传播算法进行聚类. 另外，网络嵌入(Network Embedding)[4]研究领域近年来备受关注，学者们相继提出了 DeepWalk[5]、LINE[6]和 node2Vec[7]算法. Zhang 等[8]将 Network Embedding 的思想用于同名区分，通过最大化有边相连的节点间的向量内积训练出论文的向量化表示，再采用 K-Means 算法进行聚类.

尽管目前已存在一些针对学者同名区分的研究成果，但对非常大众化的学者名(如 Wei Wang)，特别是在同一个机构存在多个同名不同人的学者，或是同名不同人的学者在同一会议发表过文章等复杂情况下，现有方法不能很好地进行同名区分，会造成精准率偏高但召回率偏低，或是召回率偏高但精准率偏低等问题. 本文提出基于网络最大流的同名区分(MFND)算法，能够有效地降低不同学者实体之间的共享特征(机构、发表会议等)给同名区分带来的影响，从而达到更好的同名区分效果.

1 问题定义

本文研究的是如何解决学者同名的区分问题. 学者同名区分指的是对包含同一个学者人名的论文进行区分，根据人名指代的真实实体不同将论文集合划分成若干子集合. 形式上，即给定一个学者人名，以及作者列表中包含该人名的论文集合 $P=\{P_1,P_2,\cdots,P_I\}$，求 P 的划分方式 $D_k=\{D_1,D_2,\cdots,D_K\}$，使得 P 为同名但不同人的真实学者实体所发表的论文集合. 某作者与论文关系图示例如图 1 所示. 其中，A_1、A_2、A_3 是同名不同人的 3 位真实作者.

图 1 中 $P_i(i=1,2,\cdots,10)$ 与 $A_m(m=1,2,3)$ 的对应关系是真实正确的. 然而在进行同名区分之前，无法得知有多少个同名的作者实体 A_m，更无法

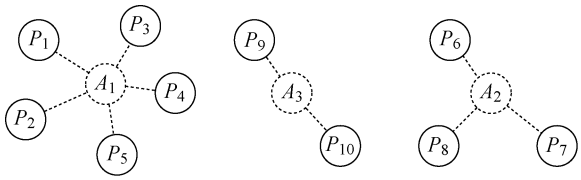

图 1 作者与论文关系图

Fig. 1 The relationship between authors and papers

知道 P_i 与 A_m 的真实对应关系，故图中采用虚线表示. 那么，同名区分的目标在于将论文节点 P_i 进行聚类，使得同一位作者所写的论文尽可能地聚到同一类中，每一类代表一个真实作者实体 A_m 所写的论文集合.

2 MFND 算法

2.1 基本思路

首先，可以把同名区分问题转化为异构网络图[9]中的聚类问题. 异构网络图即为拥有多种不同类型节点和边的网络图. 若将待区分的学者名字对应的所有论文，以及论文的合作者、所属机构等特征提取出来融合在同一网络中，可形成一张异构网络图. 例如，将图 1 中的论文及其特征融入一张图中，可形成如图 2 所示的异构网络图. 其中：$C_q(q=1,2,3,4)$ 表示合作者；$O_p(p=1,2,3)$ 表示机构；$V_r(r=1,2,3)$ 表示论文发表所在的会议 / 期刊. 在真实的聚类情况中，(P_1,P_2,P_3,P_4,P_5)，(P_6,P_7,P_8)，(P_9,P_{10}) 分别同属于一个类. 而同名区分的目标在于，在不知道真实类的情况下，如何将论文节点 P_i 进行聚类.

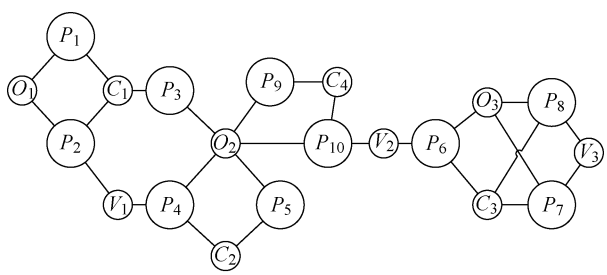

图 2 学术异构网络关系图

Fig. 2 Academic heterogeneous relationship network

聚类的关键之一在于如何计算 P_i 和 P_j 的相似度 $\mathrm{sim}(P_i,P_j)$. 由图 2 可知，P_3-P_5 和 P_9-P_{10} 都拥有一个共同特征 O_2. 而这种不同真实类之间共享的特征，会给相似度的计算带来很大的困扰. 例如，若采用随机游走的方式，则从源节点到目的节点产生的随机游走路径越多，两节点间的相似度就越高. 从 P_4 出发，可产生许多随机游走路径到达 P_2，同样

也可以产生几乎等量多的路径到达 P_9(见图 2). 这就导致基于随机游走的相似度计算方式无法很好地区分开 $\mathrm{sim}(P_4,P_2)$ 和 $\mathrm{sim}(P_4,P_9)$ 的大小差异.

基于此观察,采用基于网络最大流的方式计算相似度,可降低不同真实类之间的共享特征带来的影响. 具体而言,为了避免特征节点 $(C_q/O_p/V_r)$ 被重复用于提高相似度,给每个特征节点设定容量(初始化容量均为 1,后续步骤会采取策略更新),再计算论文节点 P_i 两两之间的最大流量. 例如在图 2 中,特征节点设定容量为 1 后,可通过计算获得 P_4-P_2 的最大流量为 2,而 P_4-P_9 的最大流量为 1,两者具有较明显的大小差异.

2.2　计算所有论文节点对的最大流

采用基于网络最大流的方式计算相似度,需计算论文节点 P_i 两两之间的最大流. 而常用的最大流算法,如 Edmonds-Karp[10] 等算法,仅用于边有容量的网络. 因此,需要将节点容量转换成边容量. 将所有带有容量限制的特征节点 $X_s(C_q/O_p/V_r)$,拆分成 X_{s_in} 和 X_{s_out} 两个节点,并添加有向边(X_{s_in},X_{s_out}),使边容量 $c(X_{s\_in},X_{s\_out})$ 等于原始节点容量 $c(X_s)$. 对于原本与 X_s 相连的每个论文节点 P_i,添加(P_i,X_{s_in}) 和(X_{s_out},P_i) 两条有向边. 将图 2 中的 C_2 节点展开为如图 3 所示的形式.

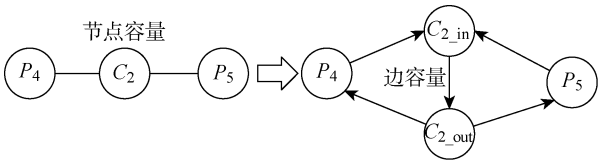

图 3　节点容量转化为边容量示意图

Fig. 3　Diagram of changing vertex capacity to edge capacity

虽然将节点容量转换成边容量后,整个网络从无向图变成了有向图,但只从 P_i 节点来看,整个网络图依然是对称的,即 P_i 到 P_j 的最大流等于 P_j 到 P_i 的最大流. 因此,可采用 Gomory-Hu 算法[11] 计算所有 P_i 节点对的最大流.

若是每两个节点都计算 1 遍最大流,则需要计算 $I(I-1)/2$ 次最大流. 而 Gomory-Hu 算法通过求解 Gomory-Hu 树就可以获得所有节点对之间的最大流,只需要计算 $I-1$ 次最大流,大大地降低了时间复杂度. Gusfield[12] 提出了一种更容易实现的 Gomory-Hu 树求解方式. 基于文献[12] 的算法思想,计算所有论文节点对的最大流. 具体算法如下.

算法 1　求解所有论文节点对的最大流

输入　$G(V,E)$,论文节点集合 P_i // G 为一个图;V 为图 G 中的顶点集合;E 为图 G 的中边集合

输出　论文节点两两之间的最大流

(1) Initial Visited←{}
(2) r←rand(P) //随机选取一个论文节点设为星型树的中心 r
(3) for each $i\in P-\{r\}$ do
(4)　Tree[i]←r//将其他论文节点与 r 相连
(5) for each $i\in P-\{r\}$ do //论文节点 i 为源节点
(6)　j←Tree[i] //j 为在星型树中与论文节点 i 相邻的节点,并将 j 作为目的节点
(7)　cut, partition←MinimumCut(i, j) //计算最小割 cut 及割集 partition
(8)　Flows[i, j]←cut, Flows[j, i]←cut //记录论文节点 i 与 j 之间的最大流
(9)　for each k in the same partition as i do //对于与论文节点 i 在同一割集的每个节点 k
(10)　if $k\in P$ and $k\notin$ Visited and Tree[k]=j //若 k 是未访问过的论文节点且与 j 相连
(11)　Tree[k]←i //在星型树中将 k 与论文节点 i 相连
(12)　for each $m\in$ Visited-$\{j\}$ do
(13)　if cut < Flows[i, m]
(14)　Flows[i, m]←cut, Flows[m, i]←cut
(15) Visited←Visited$\cup i$ //论文节点 i 标为已访问
(16) return Flows

2.3　更新特征节点容量

为了更进一步限制不同真实类之间的共享特征(见图 2 中的 O_2 和 V_2)带来的影响,可以为不同特征节点赋予不同的容量. 直观来看,若一个特征节点被越多越大的真实类所共享,其容量应该越小. 基于此想法,所设计的容量更新策略为:在得到基于初始化容量(均为 1)的两两论文节点间的最大流后,计算特征节点的所有邻居可组成多少个内部(两两节点之间)流量大于 1 的连通子图. 将连通子图内的节点个数记录为 S_l,将所有 l 个连通子图内的节点个数 s 记录为序列$[S_l]$,再基于序列$[S_l]$ 按照下式进行容量更新:

$$\mathrm{CA}=\prod_{l=1}^{L}\frac{1}{2+\lg(1+S_l)} \tag{1}$$

例如在图 2 中,O_2 的邻居有 P_3,P_4,P_5,P_9,P_{10}. 经过初始化容量 1 的最大流计算之后,这些邻居可组成(P_1,P_2,P_3,P_4,P_5),(P_9,P_{10}) 两个内部流量大于 1 的连通子图,则$[S_l]$ 序列为[3,2]. 根据式(1),O_2 的容量将被更新为 0.069 7. 同样,O_3 的容

量将被更新为 0.25，远大于 O_2 的容量. 下面给出更新特征节点容量的具体算法描述.

算法 2 更新特征节点容量

输入 $G(V,E)$，特征节点 x，论文节点间的最大流 Flows

输出 更新后的特征节点容量

```
(1) Initial Visited←{}, cap←1
(2)  Z←neighbors(x) //获取节点 i 的邻居集合
(3) for each z∈Z do
(4)  if z∉Visited
(5)   s←1, Visited←Visited∪z
(6)   for each y∈V-Visited do
(7)    if Flows[z, y]>1
(8)     s←s+1
(9)  cap←cap/[2+lb(1+s)]
(10) return cap
```

2.4 聚类策略

在获得论文节点间的最大流后，可将最大流量作为节点相似度进行层次聚类(HAC)[13]. 将每个论文节点当成不同的单个类簇，然后重复合并相似度最高的两个类簇. 基于节点间的相似度衡量类簇间的相似度有单连接(SL)、全连接(CL)以及平连接(AL)3 种方式，分别是将两个类簇的相似度定义为两个类簇中节点间相似度的最大值、最小值和平均值.

由于采用最大流量作为相似度的衡量方式，两个类簇之间的最大、最小和平均节点间最大流均相等，即采用 SL、CL、AL 3 种方式均等价. 因此，本文采用易于实现且时间复杂度更低的 SL 方式，每次合并相似度最大的两个论文节点所在的类簇，直到类簇的数量小于等于用户设定的聚类个数(K)，合并结束后每个类簇就代表一个同名不同人的真实学者所发表的论文集合. 在实验中将采用真实数据中的同名不同人的作者数作为聚类个数.

2.5 MFND 算法

综上所述，下面给出 MFND 算法的步骤总结.

(1) 提取论文特征，将特征节点与论文节点融合成一张网络关系图；

(2) 为每个特征节点设定初始化容量(均为 1)；

(3) 根据算法 1 计算论文节点两两之间的最大流量；

(4) 根据算法 2 更新特征节点容量，并再一次计算论文节点间的最大流量；

(5) 根据最大流量对论文节点进行层次聚类.

3 实验与结果分析

3.1 实验数据集

由于同名区分解决的是论文与真实作者之间的匹配问题，为了验证该算法的有效性，需要论文与作者真实匹配的数据集. 此前，Aminer 发布了一个同名区分数据集[2]，包含论文与作者的真实匹配关系. 由于该数据集中机构名与会议/期刊名未归一化，所以通过将其中的论文标题和作者名与 MAG (Microsoft Academic Graph)[14]进行匹配来构造一个新的数据集，即采用 Aminer 数据集所提供的论文与作者的匹配关系，而论文特征则用 MAG 所提供的特征. 选取匹配后的 10 个对应文章数最多的作者姓名进行实验，统计数据如表 1 所示.

表 1 数据集统计数据

Tab. 1 The statistics of dataset

作者姓名	论文数量	真实作者个数
Wen Gao	517	10
Sanjay Jain	342	5
Lei Wang	320	110
David E. Goldberg	255	2
Yu Zhang	245	71
Jing Zhang	230	84
Jim Gray	222	5
Lei Chen	206	40
Yang Wang	200	54
Bin Li	185	60

3.2 实验设置

实验采用 PairwisePrecision，PairwiseRecall 和 PairwiseF1 作为评价指标，

$$\text{PairwisePrecision}=\text{TP}/(\text{TP}+\text{FP}) \tag{2}$$

$$\text{PairwiseRecall}=\text{TP}/(\text{TP}+\text{FN}) \tag{3}$$

$$\text{PairwiseF1}=\frac{2\times\text{PairwisePrecision}\times\text{PairwiseRecall}}{\text{PairwisePrecision}+\text{PairwiseRecall}} \tag{4}$$

式中：TP 表示真实属于同一类且预测分配到同一类的对数；FP 表示真实不属于同一类但预测分配到同一类的对数；FN 表示真实属于同一类但预测分配到不同类的对数. PairwisePrecision 越高代表越多同一个类中的文章是真实属于同一人的，PairwiseRecall 越高则代表越多同一人写的文章聚到同一个类中. PairwiseF1 为 PairwisePrecision 和 PairwiseRecall 的调和平均数，PairwiseF1 越高代表综

合性能越好.

为了有效验证 MFND 算法的性能,选择以下 3 种方法进行对比.

(1) 将整个网络当成同构网络,使用 Node2Vec 算法[7] 训练得到节点的向量化表示,再分别采用 Single-Link 和 Average-Link 两种方式的层次聚类进行聚类.

(2) 采用文献[8]中公开的源码实现. 由于文献[8]中只使用了合作者的特征,为了公平起见,本文进行了两组对比实验,其中一组只使用合作者特征,而另外一组则使用合作者、机构和论文发表所在的会议/期刊 3 种特征.

(3) 获取 Aminer 官网中同名区分后的论文与作者对应关系并进行对比. 该网址的同名区分是根据文献[2]中的算法实现的.

3.3　实验结果

算法对比实验结果如表 2 和 3 所示. 表 2 为只使用合作者特征的实验结果,表 3 为使用了合作者、机构和论文发表所在的会议/期刊 3 种特征的实验结果. 其中:"Node2Vec-SL"和"Node2Vec-AL"均表示采用 Node2Vec 算法,"-SL"表示采用单连接层次聚类,"-AL"表示采用平连接层次聚类;"MFND"表示所提出的完整算法;"MFND-R"表示去掉第 4 步骤后的算法. σ 为精准率;ε 为召回率;F_1 为精确率

表 2　使用合作者特征的同名区分结果

Tab. 2　Result of name disambiguation using co-author feature

作者姓名	Node2Vec-SL			Node2Vec-AL			A4			MFND-R			MFND		
	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1
Wen Gao	92.0	96.8	94.4	93.0	93.0	93.0	98.6	28.2	43.8	100.0	89.0	94.2	100.0	89.0	94.2
Sanjay Jain	82.6	98.0	89.6	86.4	90.9	88.6	96.9	70.5	81.6	100.0	68.9	81.6	100.0	68.9	81.6
Lei Wang	35.5	73.6	47.8	65.9	60.0	62.8	84.0	70.3	76.5	90.2	70.3	79.0	90.2	70.3	79.0
David E. Goldberg	99.2	99.2	99.2	100.0	90.2	94.8	99.4	55.5	71.2	100.0	60.8	75.6	100.0	60.8	75.6
Yu Zhang	6.4	76.7	11.9	6.6	74.1	12.1	82.4	45.3	58.4	100.0	45.2	62.2	100.0	45.2	62.2
Jing Zhang	15.0	67.1	24.5	41.8	53.4	46.9	85.5	46.8	60.5	98.4	47.6	64.1	98.4	47.6	64.1
Jim Gray	95.5	94.6	95.0	97.6	60.3	74.6	95.6	32.2	48.2	100.0	42.2	59.3	100.0	42.2	59.3
Lei Chen	48.5	86.1	62.1	49.1	86.1	62.5	97.6	41.6	58.3	99.9	44.4	61.4	99.9	44.4	61.4
Yang Wang	9.5	73.2	16.8	9.7	73.2	17.1	70.4	27.3	39.3	96.1	27.3	42.5	96.1	27.3	42.5
Bin Li	45.1	70.8	55.1	81.5	69.7	75.1	95.4	66.3	78.2	99.5	74.0	84.8	99.5	74.0	84.8
AVG	52.9	83.6	59.7	63.1	75.1	62.7	90.6	48.4	61.6	98.4	56.9	70.5	98.4	56.9	70.5

表 3　使用合作者、机构、论文发表所在的会议/期刊 3 种特征的同名区分结果

Tab. 3　Result of name disambiguation using features of co-author & organization & venue of publication

Author Name	Node2Vec-SL			Node2Vec-AL			A2			A4			MFND-R			MFND		
	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1	σ	ε	F_1
Wen Gao	94.2	98.0	96.1	90.5	18.3	30.5	93.9	93.5	93.7	95.0	39.8	56.1	97.6	98.0	97.8	98.8	98.0	98.4
Sanjay Jain	83.6	99.3	90.8	83.6	99.3	90.8	100.0	27.1	42.6	84.1	99.3	91.1	100.0	99.3	99.6	100.0	99.3	99.6
Lei Wang	61.2	53.0	56.8	80.0	41.2	54.4	60.5	28.2	38.5	82.0	42.5	56.0	35.4	57.4	43.8	59.6	81.4	68.8
David E. Goldberg	98.4	98.4	98.4	99.2	97.7	98.4	98.0	96.0	97.0	99.2	97.7	98.4	100.0	95.3	97.6	100.0	95.3	97.6
Yu Zhang	62.7	53.2	57.5	82.1	51.4	63.3	62.9	30.5	41.1	81.5	48.7	60.9	46.7	64.0	54.0	57.0	62.2	59.5
Jing Zhang	32.8	36.5	34.6	79.4	41.4	54.4	73.7	34.6	47.1	78.0	39.0	52.0	47.5	59.0	52.6	72.0	57.2	63.7
Jim Gray	97.3	98.2	97.7	99.0	91.1	94.9	98.2	55.9	71.3	98.1	92.0	94.9	100.0	84.9	91.9	100.0	84.9	91.9
Lei Chen	88.2	63.1	73.6	91.4	22.2	35.8	100.0	26.4	41.8	91.0	21.8	35.1	89.8	84.8	87.2	88.9	72.5	79.9
Yang Wang	35.6	54.6	43.1	72.1	50.0	59.0	61.0	34.0	43.7	58.4	41.0	48.2	29.1	40.7	33.9	29.5	35.4	32.1
Bin Li	83.2	58.4	68.6	87.0	49.9	63.4	72.6	62.4	67.1	87.6	45.9	60.2	60.3	84.3	70.3	77.8	83.5	80.5
AVG	73.7	71.3	71.7	86.4	56.2	64.5	82.1	48.9	58.4	85.5	56.8	65.3	70.6	76.8	72.9	78.4	77.0	77.2

和召回率的调和平均数;AVG 为平均数;A2 为文献[2]中所提算法;A4 为文献[4]中所提算法.

由表 2 可知,去掉第 4 步骤后的 MFND-R 算法与采用完整步骤的 MFND 算法效果相同,均要优于其他算法,并且精准率高达 98.4%.这说明"合作者"这一特征能够很好地区分不同作者所写的文章,是否采用第 4 步骤更新容量对算法结果影响不大,但 MFND-R 和 MFND 算法的召回率偏低,说明仅使用"合作者"这一特征很难将同一个人所写的文章全部聚在一起.

将表 2 和表 3 对比来看,使用 3 种特征的同名区分效果优于只使用合作者一种特征.由表 3 可知,Node2Vec-AL、Aminer 以及 ZHANG 的算法取得了较高的精准率,但召回率较低,这说明同一个人所写的文章可能分散成多个类.采用 Single-Link 层次聚类的 Node2Vec-SL 则拥有较为均衡的精准率和召回率.而 MFND 算法同样在精准率和召回率上较为均衡,且综合性能更好.另外,对比 MFND-R 和 MFND 算法可以发现,加入"机构"和"论文发表所在的会议/期刊"特征后,采用第 4 步骤更新容量能够有效提高精准率,进而提升综合性能.

4 结语

本文提出一种基于网络最大流的同名区分(MFND)算法.该算法将论文及其特征(合作者、机构等)融合成一张网络图,根据特征节点的被共享程度设定不同的容量,再计算论文节点间的最大流量,并基于最大流量进行聚类.实验结果表明:与现有的同名区分算法相比,MFND 算法在精准率和召回率上有较为均衡的表现,综合性能更好.

参考文献:

[1] YIN X X, HAN J W, YU P S. Object distinction: Distinguishing objects with identical names[C]// **2007 IEEE 23rd International Conference on Data Engineering**. Istanbul, Turkey: IEEE, 2007: 1242-1246.

[2] TANG J, FONG A C M, WANG B, *et al*. A unified probabilistic framework for name disambiguation in digital library[J]. **IEEE Transactions on Knowledge and Data Engineering**, 2012, 24(6): 975-987.

[3] FAN X, WANG J, PU X, *et al*. On graph-based name disambiguation[J]. **Journal of Data and Information Quality**, 2011, 2(2): 10.

[4] CUI P, WANG X, PEI J, *et al*. A survey on network embedding[J]. **IEEE Transactions on Knowledge and Data Engineering**, 2019, 31(5): 833-852.

[5] PEROZZI B, AL-RFOU R, SKIENA S. Deepwalk: Online learning of social representations[C]// **Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, USA: ACM, 2014: 701-710.

[6] TANG J, QU M, WANG M Z, *et al*. Line: Large-scale information network embedding[C]// **Proceedings of the 24th International Conference on World Wide Web**. Florence, Italy: ACM, 2015: 1067-1077.

[7] GROVER A, LESKOVEC J. Node2vec: Scalable feature learning for networks[C]// **Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. San Francisco, California, USA: ACM, 2016: 855-864.

[8] ZHANG B C, HASAN M A. Name disambiguation in anonymized graphs using network embedding[C]// **Proceedings of the 2017 ACM on Conference on Information and Knowledge Management**. Singapore: ACM, 2017: 1239-1248.

[9] SUN Y Z, NORICK B, HAN J W, *et al*. PathSelClus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks [J]. **ACM Transactions on Knowledge Discovery from Data**, 2013, 7(3): 11.

[10] EDMONDS J, KARP R M. Theoretical improvements in algorithmic efficiency for network flow problems[J]. **Journal of the Association for Computing Machinery**, 1972, 19(2): 248-264.

[11] GOMORY R E, HU T C. Multi-terminal network flows[J]. **Journal of the Society for Industrial and Applied Mathematics**, 1961, 9(4): 551-570.

[12] GUSFIELD D. Very simple methods for all pairs network flow analysis[J]. **SIAM Journal on Computing**, 1990, 19(1): 143-155.

[13] WILKS D S. Statistical methods in the atmospheric sciences[M]. 3rd ed. Oxford, UK: Elsevier, 2011: 603-611.

[14] SINHA A, SHEN Z H, SONG Y, *et al*. An overview of microsoft academic service (MAS) and applications[C]// **Proceedings of the 24th International Conference on World Wide Web**. Florence, Italy: ACM, 2015: 243-246.

(本文编辑:石易文)