GitHub社交网络分析

管实1801班 U201816007 李佳妮

管实1801班 U201816041 李欣羽

管实1801班 U201815984 李博骁

2020 年 7 月 15 日

目录

1 背景简介

1.1 社交网络

社交网络即社交网络服务（Social Network Service, SNS），源自互联网社交，以网络为载体将人们连接起来，形成具有某一特点的团体，是一种典型的复杂网络。

社交网络以人为节点，节点之间的边一般代表人与人之间的社会关系。互联网的飞速发展使人们有了更便捷和多样的社交方式，不同领域、层级的人们之间的交互变得更加频繁，人们的社交范围不再受空间限制而趋于多元化，社交关系的形式越来越丰富，也因此产生了类型更多样的社交网络，涌现了众多互联网社交网站和平台，如国内的QQ、人人网、微信、微博，国外的Facebook、Instagram等。社交网络已经成为了人们获取信息、展现自我的重要窗口，是现代人们生活中不可缺少的关键部分。与交通网络、通讯网络等其他网络相比，社交网络的复杂性主要源于用户群体互动行为，包含了更加海量和多元化的信息，因此社交网络也是大数据的重要载体。由于社交网络固有的社会性，面向社交网络的研究能够帮助人们探索人类社会行为特征和规律，人类社会的结构特点、重大社会问题等。

1.2 GitHub社交网络

GitHub是一个面向开源及私有软件项目的托管平台，作为开源代码库及版本控制系统，社交化、民主化的自由风格吸引了超过900万开发者用户。每个开发者可以在自己的个人主页上发布和管理自己的代码和开发的项目，并将项目代码与他人共享，同时可以关注其他开发者的项目代码和开发动态，在他人的项目下进行评论，并star和fork自己感兴趣的项目代码。其作为开发者远程协作和交流学习的重要工具和平台，汇集了无数有趣、优质、经典的项目，是最受欢迎的开发者社区平台。与其他社交网络相比，GitHub社交网络的特殊性在于用户类型单一，大多为程序员，社交内容集中于计算机编程相关的主题。本报告以GitHub为研究对象，以开发者的项目仓库数量和star数为选取标准，得到由GitHub社区中项目经验成熟的开发者构成的社交网络，主要探究该社交网络的结构特性，包括网络模型统计特性、用户社交影响力、社区结构特性等。

2 GitHub社交网络的构建

2.1 数据收集和处理

本报告的GitHub社交网络数据集来自于kaggle，该社交网络于2019年6月从公共API中收集，数据包括289003条有向边、37701个节点，节点代表至少有10个仓库的开发者，有向边代表开发者之间的follow关系，节点属性包括开发者的名称和开发者建立的各仓库的star数量。由于原数据集的节点数和边数过于庞大，采取雪球抽样的方法，以每个节点的star总数为键，对所有节点从高到低排序，选取star最多的1000个节点为初始节点范围，设置两层搜索深度，每次搜索得到与当前范围的节点相连的新节点，加入节点范围，最终得到一个相对较小的数据集，共包含19746个节点和68144条有向边，节点ID沿用原数据集，不做重新编号。

本报告中的数据计算采用Python和Gephi实现，网络的可视化采用Gephi实现。

2.2 数据和网络概况

对处理得到的数据集进行描述性统计：

表 1: 数据描述性统计

统计量	stars	入度（followers）	出度（follows）
count	19746.000000	19746.000000	19746.000000
mean	36362.765877	3.451028	3.451028
std	8848.967602	8.721536	60.596419
min	9657.000000	0.000000	0.000000
25%	30244.750000	1.000000	0.000000
50%	35167.000000	2.000000	0.000000
75%	41050.750000	3.000000	0.000000
max	86860.000000	628.000000	6809.000000

其中，入度与出度分别表示开发者的粉丝数与关注数。根据描述性统计结果，雪球抽样产生的网络中大多数开发者的stars数都较大。对于关注数而言，其方差较大，大部分开发者关注数都为0，仅有少数开发者关注了较多其他开发者。

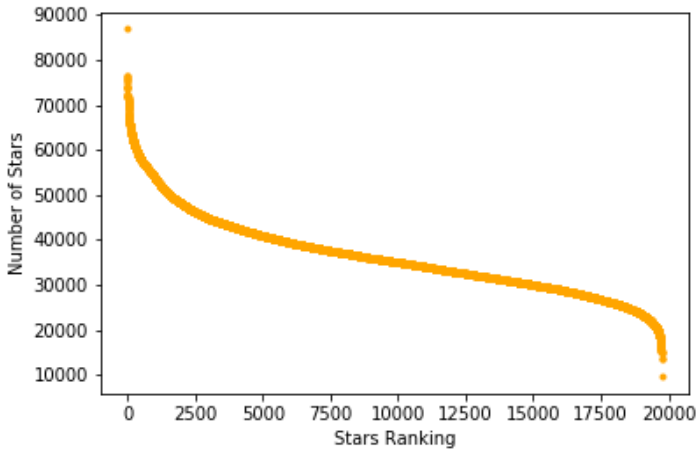


图 1: 节点Stars降序分布

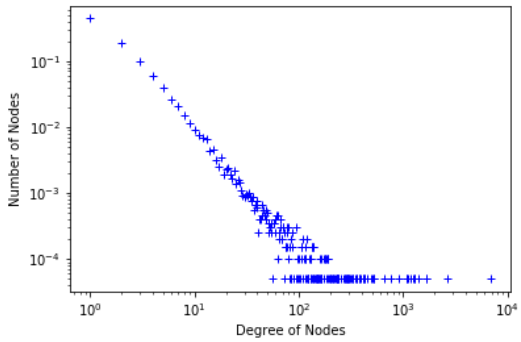
3 网络模型统计特性

3.1 度分布——无标度特性

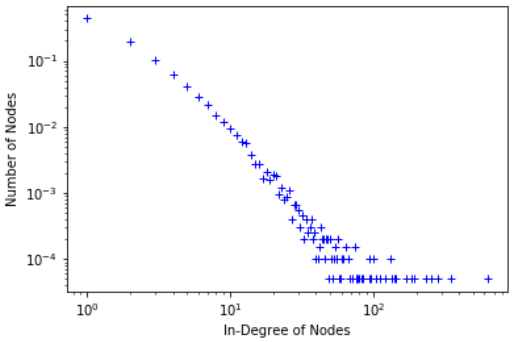
无标度特性（scale-free property）是描述大量复杂系统整体上严重不均匀分布的一种内在性质，具体指的是网络的度分布符合幂律分布，即$P(k) \sim k^{-\gamma}$，网络中少数的节点拥有极其多的链接，大

多数节点只有很少量的链接，这样的网络被称为无标度网络。研究表明，许多社交网络具有无标度特性 [1]，绝大多数用户拥有较少的社会关系，小部分用户拥有较多的社会关系。本部分通过分析GitHub社交网络的度分布，来验证该网络是否具有无标度特性。

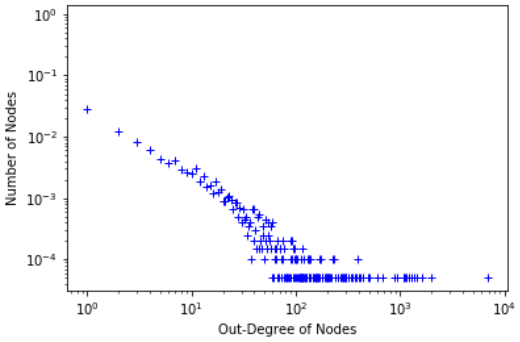
用Python的networkx库计算各节点的度（入度和出度的和）、入度和出度，分别以这三个指标为X轴，各度数对应的节点数为Y轴，绘制双对数坐标图，如下图所示。



(a) 度分布



(b) 入度分布



(c) 出度分布

图 2

可以发现，节点的度、入度和出度分布曲线均近似为直线，说明三者的分布均符合幂律分布，大多数开发者的度数很低，而极少数的开发者度数很高。从具体的数值来看，超过一半的节点的度为1和2，大多数节点的出度和入度为0和1，相比度数最多的节点差距十分之大。

表 2: 排名前十的度数

排名	度	出度	入度
1	6813	6809	628
2	2616	1988	351
3	1678	1610	284
4	1653	1459	257
5	1397	1378	232
6	1297	1295	194
7	1274	1224	184
8	1233	1149	171
9	1171	1111	143
10	1121	1085	140

综合上述对该网络的度分布的分析，可以得到，该GitHub网络具有无标度特性，是一个无标度网络。

3.2 小世界现象

小世界现象（small-world phenomenon）指的是网络具有较小的平均路径长度和较大的平均聚集系数。研究表明，大多数社交网络存在小世界现象 [1]，网络直径较小，平均路径长度短，平均聚类系数相比随机网络的聚类系数要大得多。下表展示了国外的几个大型社交网络的平均路径长度、网络直径、聚类系数及其与ER随机网络的聚类系数比值。本部分通过计算该GitHub社交网络的平均路径长度和平均聚集系数，分析这两个指标值，来判断该网络是否存在小世界现象。

表 3: 国外社交网络的相关指标 [1]

社交网络	平均路径长度	网络直径	平均聚类系数	和ER随机图的聚类系数之比
Flickr	5.67	13	0.313	47200
LiveJournal	5.88	12	0.330	119000
Orkut	4.25	6	0.171	7240
YouTube	5.10	13	0.136	36900

利用Gephi计算，可以得到网络的平均路径长度为3.128，网络直径为11，说明虽然网络节点数量庞大，但大多数开发者相互可以在四步之内建立联系，符合六度分隔理论。平均聚集系数为0.114，而一般情况下ER随机网络的平均聚类系数的数量级在$10^{-6} - 10^{-5}$，与前者相差很大，说明GitHub社交网络具有小世界现象。

3.3 连通性

网络的连通性（connectivity）反映了网络中节点的连接程度，一个网络的连通分量越少，说明这个网络连接完整性越好，反之表明网络越破碎，存在越多的独立团体，导致网络的信息流通性越差。

我们用python计算了该GitHub网络的弱连通分量数目，结果显示共有7个弱连通分量，具体如下表所示。

表 4: GitHub社交网络的弱连通分量及其节点数

连通分量序号	包含节点数	节点比例	包含节点的ID
1	19733	99.93%	···
2	3		11247, 13424, 36848
3	2		7375, 33661
4	2	0.07%	14013, 36501
5	2		6247, 17925
6	2		22132, 33504
7	2		34047, 37315

可以发现最大的连通分量包含的节点数占了全网络的99.93%，说明该网络基本是连接起来的，说明GitHub社区里具有丰富项目经验的“高端”开发者之间是相互联系的。只有少数几个只包含两至三个节点的连通分量独立于网络主体，这部分游离的节点仅占总数的0.07%。表4列出了这些节点的stars数，可以发现这些用户的stars数量相比整体的stars分布，处在较高的水平，说明极小部分的“大牛”开发者比较特立独行，不关注别人，也少有其他同层级的开发者关注。

表 5: 部分连通分量包含的节点

连通分量序号	包含的节点ID	star数
2	11247	43303
	13424	62921
	36848	44889
3	7375	56395
	33661	50119
4	14013	59680
	36501	33123
5	6247	44189
	17625	57136
6	22132	41880
	33504	56428
7	34047	61794
	37315	47497

然而，该网络的强连通分量却极多，接近节点总数，说明其中的双向关系很少，基本都是单向关系，反映了这些开发者之间的关注不像一般社交网络中那么紧密。

3.4 同配性

网络的同配性（assortativity）反映了网络中度相近的节点之间相互关联的程度，同配系数为正说明网络中的节点更倾向于连接相似度的节点，为负说明节点会连接一些度数和自己相差较大的节点，这样的网络被称为异配网络。

我们利用python计算得到该GitHub网络的同配系数为-0133，说明其具有异配性，一般开发者会倾向于选择关注者数量更多的其他开发者“大V”用户，这些名人用户会有极其多的关注者，随之有更大的影响力，会吸引越来越多的用户的关注，从而使网络呈现出异配性。

4 影响力分析

影响力分析是社交网络分析最重要和核心的内容之一，包括影响力度量、影响力动态演化、影响力传播等问题。社交影响力通过人们的交互行为体现，网络中影响力最大的节点和团体往往对其他节点乃至网络结构起核心作用和影响，从GitHub平台的现实意义来说，可能代表这个社区中最有经验、有权威的开发者及其团队，在计算机编程的某一方向有较深的项目造诣，开发的项目具有很高的现实应用意义或学术创新意义；或者代表最活跃的开发者们，热衷于与大量开发者合作交流。分析GitHub社交网络中的最富影响力的用户，一定程度上有利于帮助用户们快速把握目前自己研究领域中最热点、最前沿的研究和应用，以及领域内的顶尖开发者前辈，便于更高效的学习和交流。

目前，对社交网络影响力的评估的方法包括基于节点自身属性的方法、基于概率模型的分析方法以及基于链接分析的方法。其中，基于节点自身属性的方法主要通过网络中各节点的中心度的测定来实现，主要包括点度中心度、紧密中心度与中介中心度等中心度指标。基于链接分析的方法则考虑到社交网络的动态性，利用网页链接分析的原理来对节点的影响力、重要性进行排序，弥补了前者未考虑社交网络复杂性的缺陷。

本部分将从基本的中心性指标出发，分析Github开发者网络中各开发者的影响力，进一步结合GitHub用户的stars属性，使用链接分析中的HITS算法，找出其中最具影响力的开发者节点。

4.1 中心性指标

4.1.1 点度中心度

点度中心度衡量节点对其邻居的平均影响力。

对于有向图，需要分别计算点入度中心度与点出度中心度。利用networkx分别计算各节点的相对点入度/点出度中心度，得到如下结果：

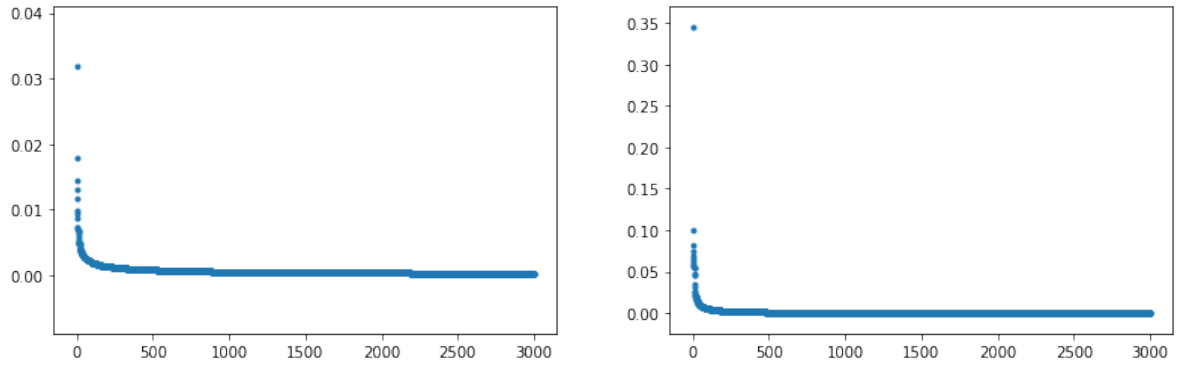
表 6: 点入度中心度排名前十的节点

ID	排名	点入度中心度	出度	入度
31890	1	0.0318	1988	628
35773	2	0.0178	923	351
36628	3	0.0144	30	284
36652	4	0.0130	58	257
35008	5	0.0117	99	232
19222	6	0.0098	1459	194
15191	7	0.0093	117	184
18163	8	0.0087	66	171
36790	9	0.0072	11	143
33029	10	0.0071	94	140

以降序排名为X轴，相应指标值为Y轴，对应的总体分布如下：

表 7: 点出度中心度排名前十的节点

ID	排名	点出度中心度	出度	入度
27803	1	0.3448	6809	628
31890	2	0.1007	1988	351
13638	3	0.0815	1601	68
19222	4	0.0739	1459	194
9051	5	0.0698	1378	19
2078	6	0.0656	1295	2
7027	7	0.0620	1224	9
10001	8	0.0582	1149	22
5629	9	0.0562	1111	10
73	10	0.0550	1085	6



(a) 入度中心度分布 (b) 出度中心度分布

图 3: 入度中心度和出度中心度

由于点入度中心度与入度直接相关，对应入度较大者，其点入度中心度也相对较高；点出度中心度同理。

点入度中心度表现一个人的被关注程度，点入度中心度较高者表明其更受近邻节点的重视，在该开发者网络中可以相应的认为入度中心度更高者具有更高的声望，即项目和代码质量更受到大家欢迎和认可的开发者。因此，点入度中心度可以作为评价开发者影响力的一个备选指标。

点出度中心度则表现一个人关注他人的程度，点出度中心度较高者更希望与他人取得关联。在该开发者网络中可以解释为出度中心度更高者更活跃，他们更倾向于去关注别人的项目，他们是热衷于向他人学习的开发者。

同时根据整体分布图象可知，网络整体的出度中心度要高于入度中心度，这符合开发者网络中的常识，大多数人更倾向于去关注别人进行学习，而实际能够得到大量关注的开发者并不多。

4.1.2 紧密中心度

紧密中心度衡量当前节点对其他节点的间接影响力，也可间接度量社交网络中该用户的社会关系强度或是用来考察一个节点在传播信息时对其他节点的依靠程度。

利用networkx分别计算各节点的紧密中心度，得到如下结果：

表 8: 紧密中心度排名前十的节点

ID	排名	紧密中心度
36652	1	0.0473
36628	2	0.0469
37471	3	0.0467
36819	4	0.0443
15940	5	0.0434
31890	6	0.0420
36909	7	0.0418
35008	8	0.0414
37107	9	0.0410
37557	10	0.0405

以降序排名为X轴，相应排名的紧密中心度为Y轴，总体分布如下：

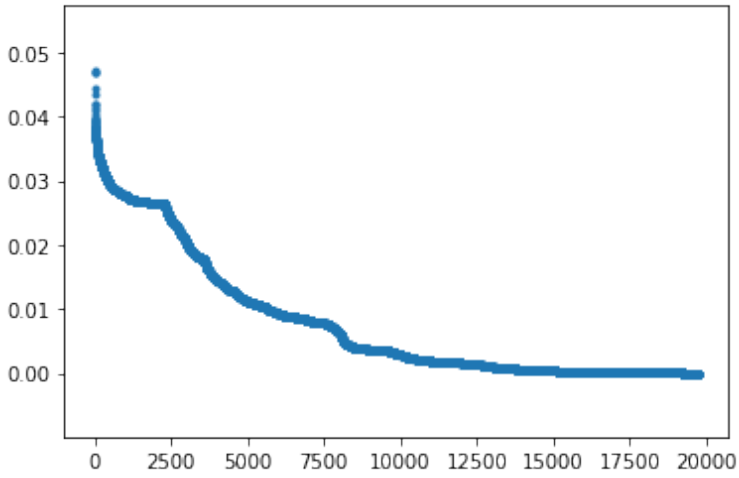


图 4: 紧密中心度分布

4.1.3 中介中心度

中介中心度衡量节点在网络中所处位置的重要性。常常用于代表其在信息传递过程中具有的影响力。

表 9: 中介中心度排名前十的节点

ID	排名	中介中心度
31890	1	0.00730
19222	2	0.00327
35773	3	0.00248
13638	4	0.00125
19253	5	0.00087
22881	6	0.00085
25477	7	0.00075
10001	8	0.00065
3712	9	0.00051
23664	10	0.00049

总体分布如下：

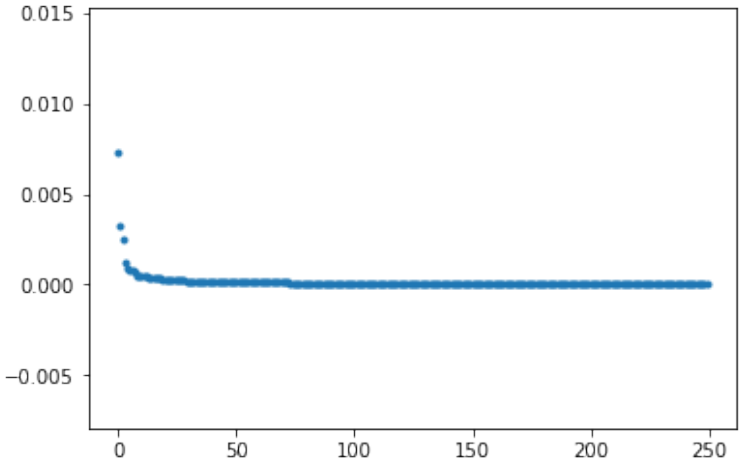


图 5: 中介中心度分布

根据分布图可以看到，网络中各节点中介中心度均较低，表明该网络相对来说各节点“权力”均等，信息在网络中的传递不会受到某些节点的很大影响。

4.1.4 聚类系数

聚类系数衡量社交网络的聚集性，即我关注或关注我的人是否互相关注。

分别计算各节点的聚类系数，绘制总体分布图：

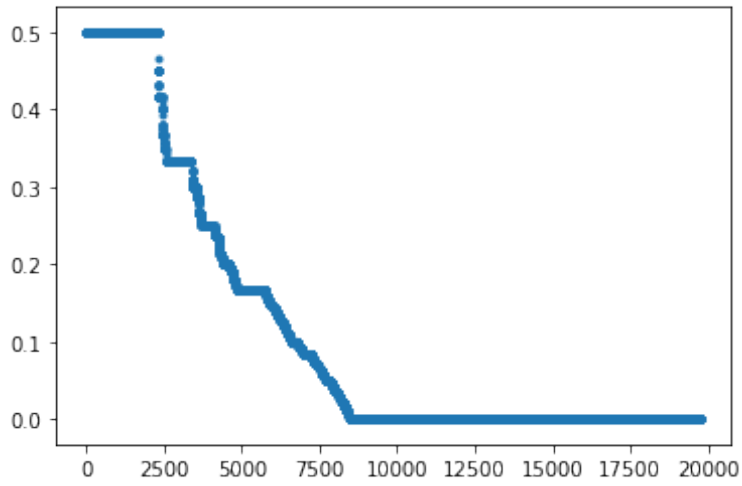


图 6: 聚类系数分布

可以看到各节点的聚类系数均在0.5以下，且约有一半多的节点的聚集系数为0，反映出该网络中的“抱团现象”并不明显，网络的结构相对松散。

4.2 基于链接分析的方法——HITS

在社交网络中，用户在网络中的影响力的评估既应考虑用户本身质量的高低，也应考虑到其影响其它用户能力的大小。单单从节点自身属性出发来评估节点在网络中的影响力往往过于单一，没有考虑到线上社交网络的交互性与动态性。因此，广大研究者开始使用基于链接分析的方法来分析社交网络节点的影响力。

链接分析原先主要用于对网页进行重要性排序。随着各种学科的交叉研究，很多研究者在评估社交网络中节点影响力时也采用了基于链接分析的PageRank算法和HITS算法。在对Github开发者网络的影响力分析中，我们采用了改进的HITS算法来对开发者的影响力进行量化与排序。

4.2.1 HITS算法

HITS算法是子集传播算法的代表算法。在HITS算法中，分为Hub页面和Authority页面，Authority页面是指与某个领域或者某个话题相关的高质量页面，Hub页面则是包含很多指向高质量Authority页面链接的网页，比如，hao123首页就是一个典型的高质量Hub页。

Hub页和Authority页之间是相互增强的关系，HITS算法基于两个基本假设：

1. 一个好的Authority页面会被很多Hub页面指向。
2. 一个好的Hub页面会指向很多好的Authority页面。

HITS利用上述两个基本假设以及相互增强关系等原则进行多轮迭代计算，每轮迭代计算更新每个页面的两个权值，直到权值稳定不再发生明显的变化为止。

4.2.2 基于改进HITS算法进行影响力分析

相比中心度指标，HITS算法分别衡量了网页本身的质量和其在网络中的枢纽作用，将用户类比

为网页，我们可以利用该算法对社交网络进行影响力分析。将每个用户都看作HITS算法中的一个网页，用户之间的相互关注就相当于网页之间的链接关系，算法收敛时用户的Authority值就可以作为社交网络中用户影响力的量化值。

基本假设 在Github社交网络中使用HITS算法的基本假设如下：

- 一个高Authority用户会被很多高Hub用户所follow。
- 一个高Hub用户会follow很多高Authority的用户。

算法原理 为了得到每个用户的Hub值和Authority值，我们用以下方法确定：

- 用户Hub值等于所有它follow的用户的Authority值之和。
- 用户Authority值等于所有follow它的用户的Hub值之和。

改进 但基本的 HITS 算法往往过于依赖节点的入度与出度，在社交网络中，关注数与被关注数往往不能很好地反映用户自身质量（可能会出现“僵尸粉”现象，关注者在网络中的互动活动很少，基本不能看作对网络有作用的有效节点）。和网页排序不同，Github用户有stars属性，能够更好地反应用户的权威性，因此我们以该值作为各节点的初始值，对原始的HITS算法进行一定的改进。

具体流程

1. 设定迭代次数k
2. 每个节点按stars数赋初始Hub值和Authority值
3. 重复k次：
    - 更新Hub值：每个节点的Hub值=∑(该节点指向节点的Authority值)
    - 更新Authority值：每个节点的Authority值=∑(指向该节点的节点的Hub值)
    - 标准化各节点的两个得分

结果分析 分别利用改进前和改进后的HITS算法进行求解。在求解过程中，我们发现尽管作了将各开发者的stars数设置为初值的改进，但改进前后得到的结果差距非常小，说明初值不同对于HITS算法最终的收敛结果影响并不大，改进并没有起到明显的效果。

由于改进前后hub和authority值基本一致，因此下面只给出了改进后的HITS算法的输出结果（按Authority进行排序）：

ID	排名	Hub值	Authority值	出度	入度	star数
31890	1	0.017006	0.000537	1988	628	27018
36628	2	0.000403	0.000480	30	284	44159
35773	3	0.010173	0.000436	923	351	40400
19222	4	0.018365	0.000394	1459	194	40230
37471	5	4.27E-05	0.000334	6	112	39831
36652	6	0.000646	0.000331	58	257	36116
35008	7	0.001087	0.000328	99	232	34516
30002	8	0.003466	0.000328	236	122	42150
22881	9	0.005323	0.000326	393	95	31152
3712	10	0.005564	0.000323	380	136	26035

从结果可以看出，由于HITS考虑了用户自身的质量，影响力前10名者不再与入度与出度极度相关，其结果相比直接使用中心度指标进行影响力分析有了一定的改进，这说明HITS算法得到的Authority值可以作为评价开发者影响力的一个不错的指标。

然而，将开发者获得的总star数纳入影响力的考虑范围并没有明显改善HITS算法的输出结果。这说明改进存在一定的问题，需要考虑其他方式来改善目前HITS算法的缺陷，如在更新Hub值和Authority值时考虑根据节点的stars数进行加权赋值等。

5 社区检测

社区结构是社交网络的重要结构特征之一，大多数社交网络中都存在很多社区群体，这些群体可能是因为兴趣爱好、职业领域、共同好友等因素而凝聚起来的。社区结构的特点在于同一社区内的节点之间连接很紧密，而不同社区之间的连接比较稀疏。目前社区结构没有严格的统一定义，可以根据不同社交网络的特点而针对性地设计算法，划分网络中的社区。

该部分利用Gephi中的模块化算法对该GitHub数据进行社区结构分析，当解析度为标准解析度时模块化指数较高，为0.401，并得到以下19个社区及相应包含的节点数。

社区序号	节点数	节点比例	包含的节点ID
1	3116	15.78%	
2	2789	14.12%	
3	2614	13.24%	
4	2290	11.60%	
5	2229	11.29%	
6	1866	9.45%	
7	1349	6.83%	
8	1176	5.96%	
9	1052	5.33%	
10	472	2.39%	
11	310	1.57%	
12	266	1.35%	
13	204	1.03%	
14	3	0.02%	11247,13424,36848
15	2	0.01%	6247,17925
16	2	0.01%	7375,33661
17	2	0.01%	22132,33504
18	2	0.01%	14013,36501
19	2	0.01%	34047,37315

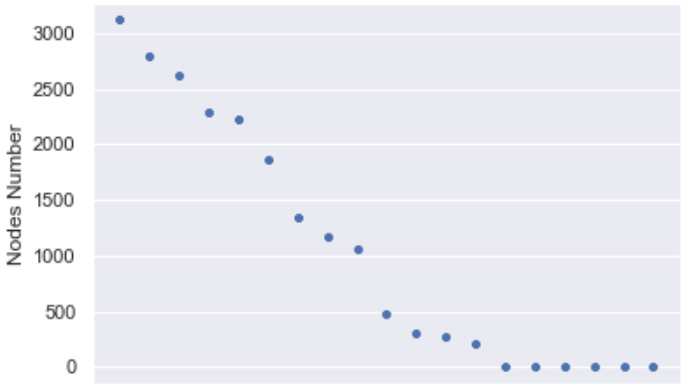


图 7: 社区节点数分布

可以发现，节点数多的社团相比节点数少的要少，按数量排序之后社区的节点数平稳下降。三分之二的社区节点数以百为量级，剩下的都是只有几个节点的游离社区。这说明GitHub的高端开发者社交网络存在较多社区。由于缺乏其他信息，该模块化结果基于社区的节点和边生成，无法对每

个社区的开发者特征进行进一步分析。可以考虑通过收集各开发者的项目主题、职业特征、研究方向等信息来设计相应的针对性的算法，获得更精准的模块化结果，来发现GitHub社交网络中各社区的群体特征和聚类规律。

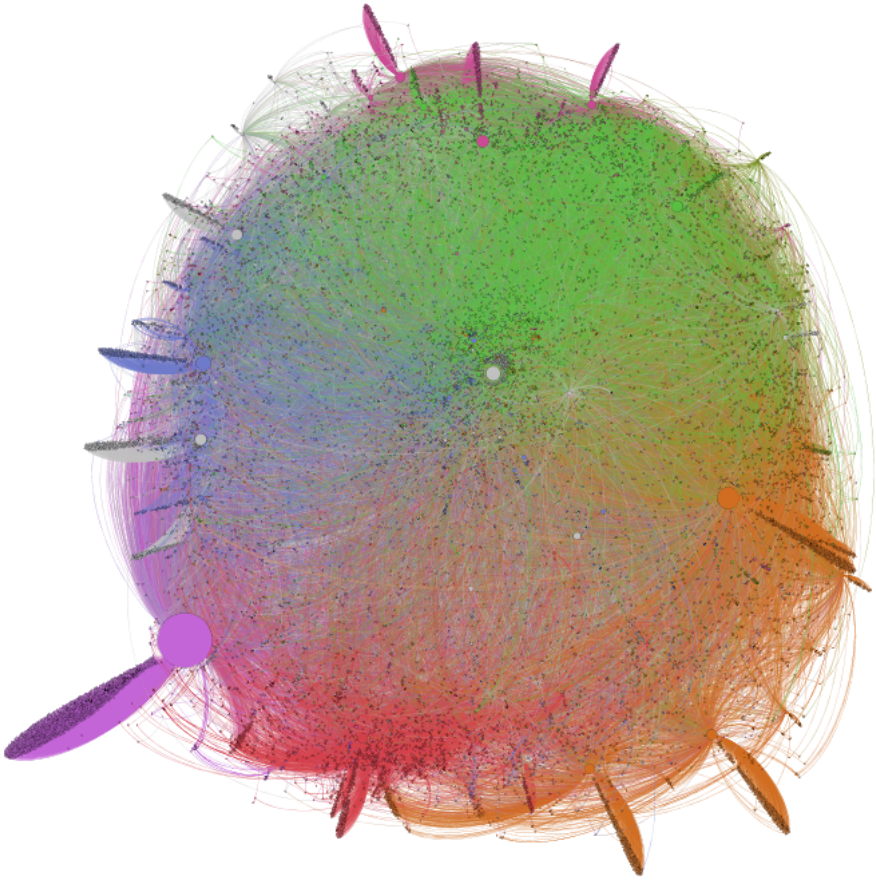


图 8: 模块化渲染的网络图

6 总结

本报告从网络模型统计特性、用户社交影响力、社区结构等方面对GitHub社交网络进行了量化分析。

在网络模型统计特性上，我们计算了相关指标，主要得到了以下结论：

1. 与大多数社交网络的特点类似，GitHub社交网络中节点的度分布符合幂律分布，具有无标度特性。

2. GitHub社交网络的平均路径长度短，符合六度分离理论，且平均聚类系数相比ER随机网络大很多，具有小世界现象。

3. 该GitHub社交网络连接完整性较好，仅有极小部分的节点游离在网络之外，但强连通分量很多说明紧密性不如一般社交网络。

4. GitHub社交网络具有异配性，开发者更倾向于关注度数较高，即关注者数量多的开发者。

此外，针对该Github网络，我们进一步分析了各开发者在该网络中的影响力。影响力分析主要从两个方面进行：基于节点自身属性的方法与基于链接分析的方法。首先，我们从节点自身属性与网络结构出发，计算并分析了各节点的点度中心度、聚集中心度、中介中心度与聚集系数，将点入度中心度作为反映节点重要性与权威性的候选评价指标。其次，考虑到中心度指标仅仅停留在节点自身属性，忽略了网络的动态性与复杂性，我们进一步选取了基于链接分析的影响力算法——HITS来评价节点的重要性，得到了该开发者网络中影响力TOP10的Github开发者。由于社交网络中入度出度并不能很好地代表用户自身的质量，我们对HITS算法作了一定的改进，选择将开发者仓库的总star数作为算法初始值以更好地评估节点自身质量。我们发现，尽管该改进改变了HITS算法的初始值，却并没有对结果产生很大的影响。这说明改进存在一定的问题，需要考虑其他方式来改善目前HITS算法的缺陷。

在社区结构上，利用Gephi的模块化算法，GitHub被划分为了19个社区，其中大部分社区节点数的数量级为$10^2 - 10^3$，反映了GitHub的高端开发者社交网络存在较多社区。

7 团队分工

- 李佳妮：收集、处理数据，文献搜集，网络模型统计特性代码编写和分析，图表可视化，报告撰写。
- 李欣羽：文献搜集，影响力部分代码编写和分析，图表可视化，报告撰写。
- 李博骁：HITS算法和社区检测部分代码编写和分析，报告撰写。

参考文献

[1] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42, 2007.