「经典重温」图表示学习经典算法 node2vec

原创 Zhihong Deng 图与推荐 2020-05-03

16KDD node2vec

image-20200502131605020

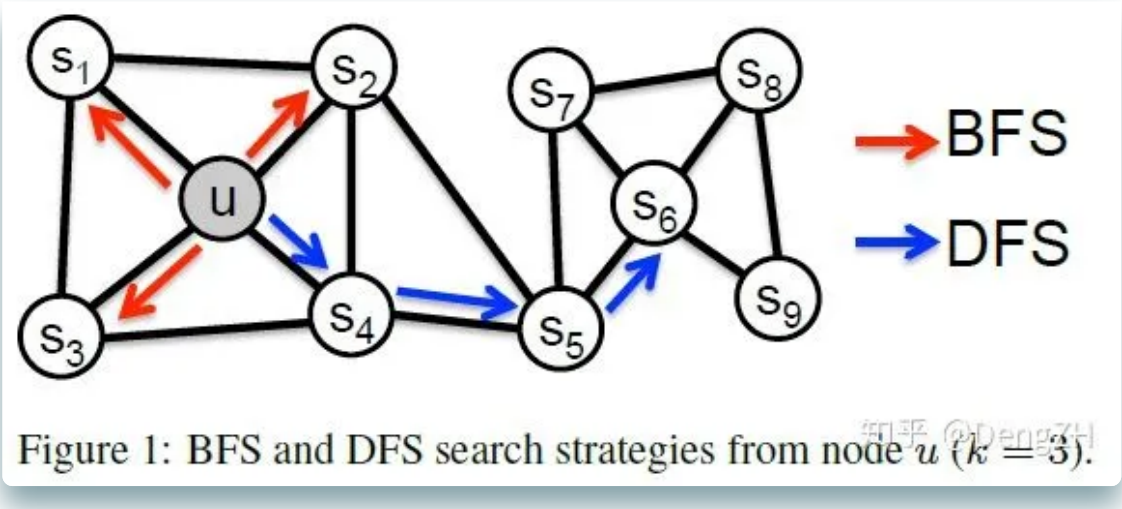
node2vec 是斯坦福男神教授 Jure Leskovec 的代表作之一，网上有非常多关于这篇论文的讨论和解析，所以这里我不再累述。

node2vec 中提出的网络的 “**同质性**” 和 “**结构性**” 是两个比较抽象的概念，之前看论文的时候没有仔细斟酌，但看了王喆大佬的文章之后，惊觉一直以来对 node2vec 理解有误。为了搞清楚这两个概念，我写了一份简单的 node2vec 代码，并进行了一些初步的探索。这篇文章的目的是要理清楚两个问题：

- 到底什么是网络的同质性？什么是网络的结构性？
- DFS 是否擅长刻画同质性，BFS 是否擅长刻画结构性？为什么？

以下开始分析：

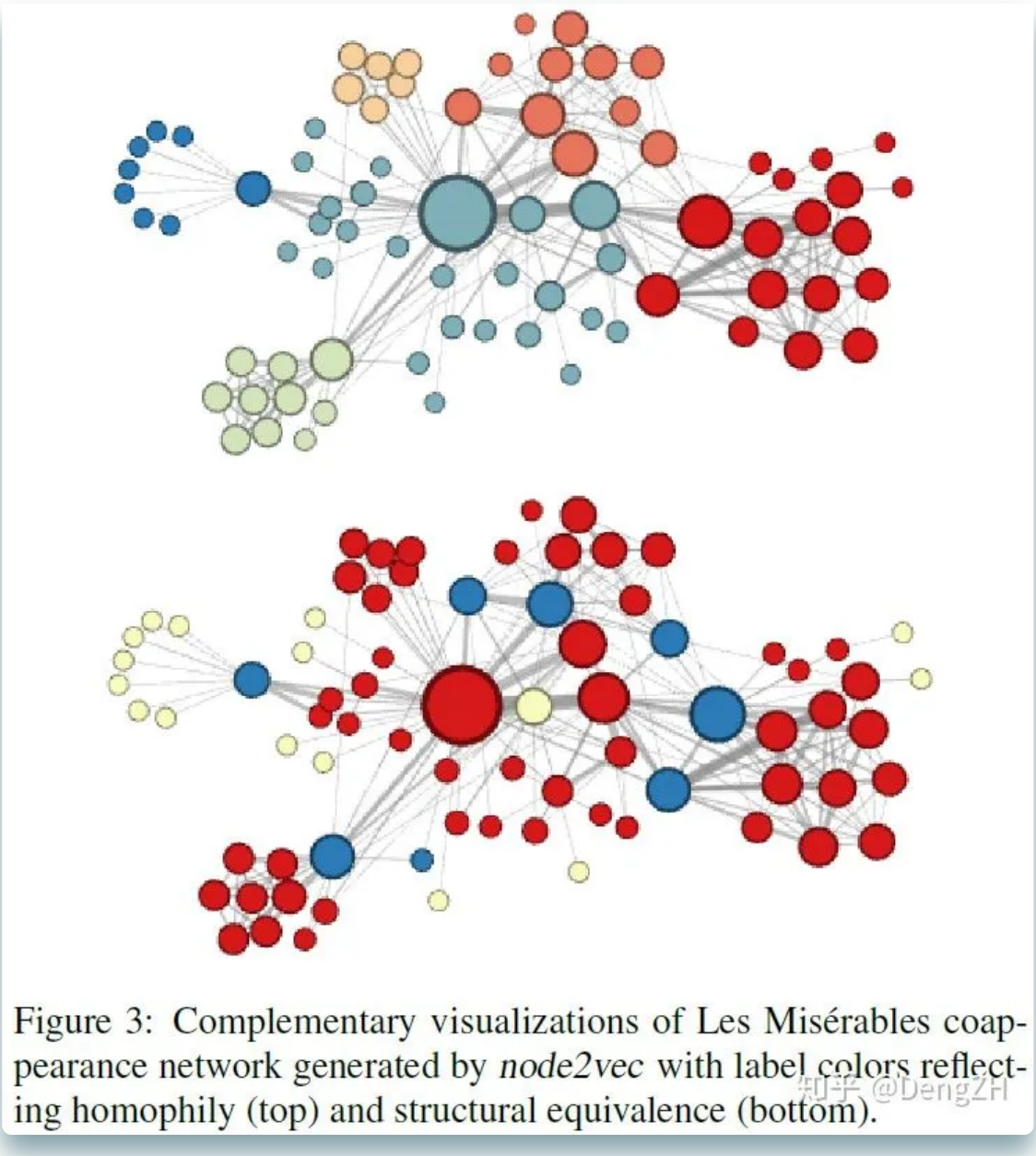
什么是网络的同质性？什么是网络的结构性？



img

直觉上，我们认为同质性是指微观上，站在结点上来看，相邻的结点应该比较相似，那么 BFS 这种更强调 1 阶邻居的游走方式应该更能表达同质性（比如上图的结点 u 和相邻的 s1，s2，s3，s4）；结构性是指宏观上，俯视整个网络，有着类似连接方式的结点应该比较相似，那么 DFS 这种能探索得更远得游走方式应该对学习结构性更有帮助（比如上图的结点 u 和结点 s6）。

但事实上，论文中给出的结论却是 **DFS擅长学习网络的同质性，BFS擅长学习网络的结构性**。从论文里的 Figure 3 中我们可以直观地进行观察：



img

图的上半部分是倾向于 DFS（p=1，q=0.5）的，可以看到，这种方式得到的 embedding 似乎有很好的聚类性质，注意这里要看结点之间的连接而不是在2D平面上的距离，每个簇的边界结点跟内部的联系要比跟外部的联系更多一些。作者认为这反映了网络的同质性；

图的下半部分是倾向于 BFS（p=1，q=2.0），一个很明显的不同就是，这种方式得到的 embedding 似乎是按功能划分的，处于 graph 边缘的结点（黄色）有类似的 embedding，连接 graph 边缘和中心的结点（蓝色，在上半部分中作为簇边界的结点）有类似的 embedding，这些结点并不都是互相连接的，但是 node2vec 得到的 embedding 仍然能学习出这样的信息。作者认为这反映了网络的结构性。

通过这个图，我们再思考一下同质性和结构性的含义，就会发现和直觉上的含义不同了。同质性并不是一个微观上的性质，作者说的**同质性是能模型能找出每个簇的边界，使得簇内结点彼此联系的紧密程度要超过跟簇外结点的联系**，这就要求模型有更大的感受野，DFS 这种能跳出局部的方式就很适合这个要求。

结构性就比较让人疑惑了，Figure 3 给出的关于结构性的表达似乎和我们直觉上差异不大，有着类似连接方式的结点会更相似。但是，BFS竟然能做到这一点？那些 embedding 相似的结点甚至并不相互连接，BFS为什么能有这种效果呢？

这里先给出后面做完实验后，感觉比较合适的一个解释。作者说的结构性并不是宏观上有相似的连接方式，而是指能够**充分学习微观上的局部结构**。比方说结点处于一个三角形连接的内部（很多论文会称之为motif），BFS 会加强对这个三角形的感知，而 DFS 则容易通过连向外界的边跳出去，所以 BFS 对局部结构得学习会比 DFS 好，这也符合对 Figure 3 的观察。但是，这并不能解释 Figure 3 中按功能划分结点这个现象，我的结论是：这种现象只能在合适的数据上，在合适的超参设定下被观察到。

DFS 是否擅长刻画同质性，BFS 是否擅长刻画结构性？为什么？

前面通过 Figure 3 来重新认识了同质性和结构性。但为什么 DFS 会擅长同质性，BFS 会擅长结构性呢？这就得再回顾一下 Figure 2，了解一下 DFS 和 BFS 到底做了什么：

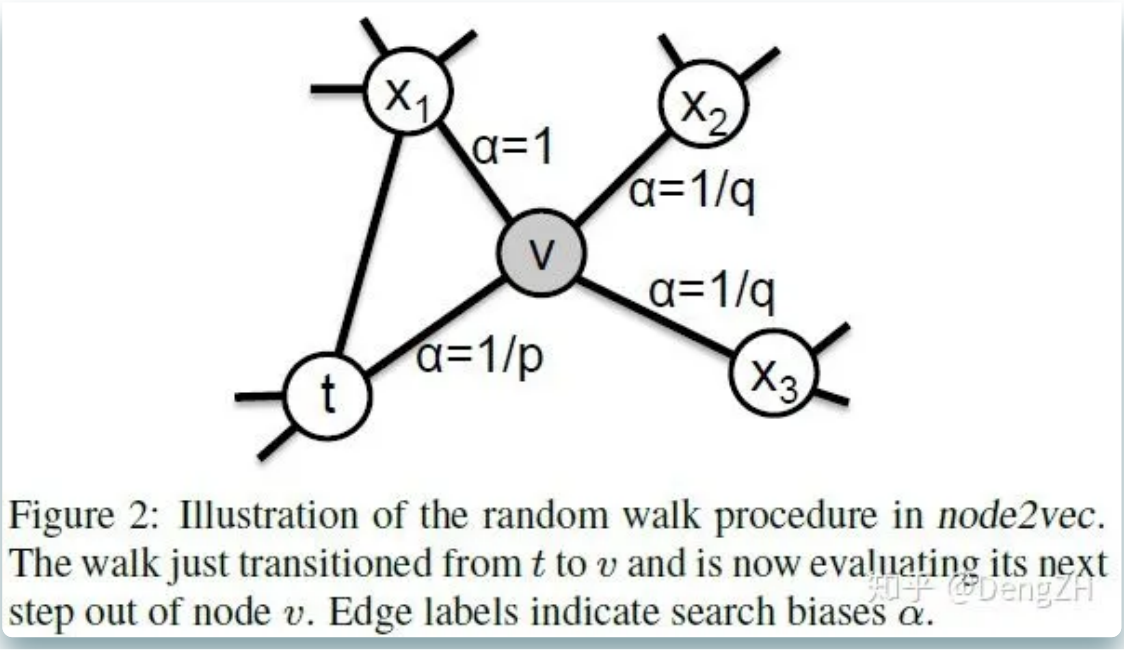


Figure 2: Illustration of the random walk procedure in *node2vec*. The walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α.

img

图中展示的是一次随机游走的中间过程，当前处于结点 v 上，上一步是从结点 t 到结点 v。

- x1 为结点 v 和结点 t 的共同邻居，设置边 v->x1 的权重为 1；
- t 为前序结点，设置边 v-> t 的权重为返回参数 p：
  - p > 1 则下一步倾向于访问共同邻居；
  - p < 1 则下一步倾向于回到前序结点。
- x2 和 x3 是结点 v 的其他一阶邻居结点，设置边的权重为进出参数 q：
  - q > 1 则下一步倾向于访问共同邻居；
  - q < 1 则下一步倾向于访问其他一阶邻居结点。

通过 p 和 q 这两个参数就可以调整游走的策略从而实现 DFS 或者 BFS。在 node2vec 中：

- DFS 是 p=1，q=0.5，
  - 此时：P(访问其他一阶邻居结点) > P(返回前序结点)=P（访问共同邻居）
- BFS 是 p=1，q=2.0，
  - 此时：P(访问其他一阶邻居结点) < P(返回前序结点)=P（访问共同邻居）

不妨在想象中检查一下，如果 P(访问其他一阶邻居结点) > P(返回前序结点)=P（访问共同邻居），那么随机游走就有可能一路推进不同的结点，构成一条重复结点较少的路径，确实符合 DFS 的理念。而如果 P(访问其他一阶邻居结点) < P(返回前序结点)=P（访问共同邻居），那么随机游走就有可能在一个较小的连接密集的局部中来会跳，构成一条重复结点较多的路径，这符合 BFS 的理念。

在得到随机游走的路径后，node2vec 就会把结点看作词，像 word2vec 学习词向量那样学习每个结点的 embedding 了。一般会采用 Skip-Gram 模式，也即使用中心词预测上下文，但无论是用 CBOW 还是 Skip-Gram，本质上都是假设一个词应该跟它所在句子的上下文词关系最密切（最相似），这也是我们理解 DFS 和 BFS 不同的关键。

如果随机游走侧重于 DFS，那么中心结点的上下文就可能同时包含不同阶的邻居；如果随机游走侧重于 BFS，那么中心结点的上下文就可能只包含有共同邻居的1阶邻居。因此，侧重于 DFS 的话，即使两个结点不彼此相连，只要它们有共同的1阶2阶邻居，也会得到相似的上下文，从而学到的 embedding 会比较像。这符合我们前面对同质性的分析，具备这种特质的 DFS 可以更好地找到簇的边界。

而侧重于 BFS 的话，处于同一个密集连接的局部的结点会更加相似，因为它们的上下文会有更多的重叠。这符合我们前面对结构性的分析，具备这种性质的 BFS 可以

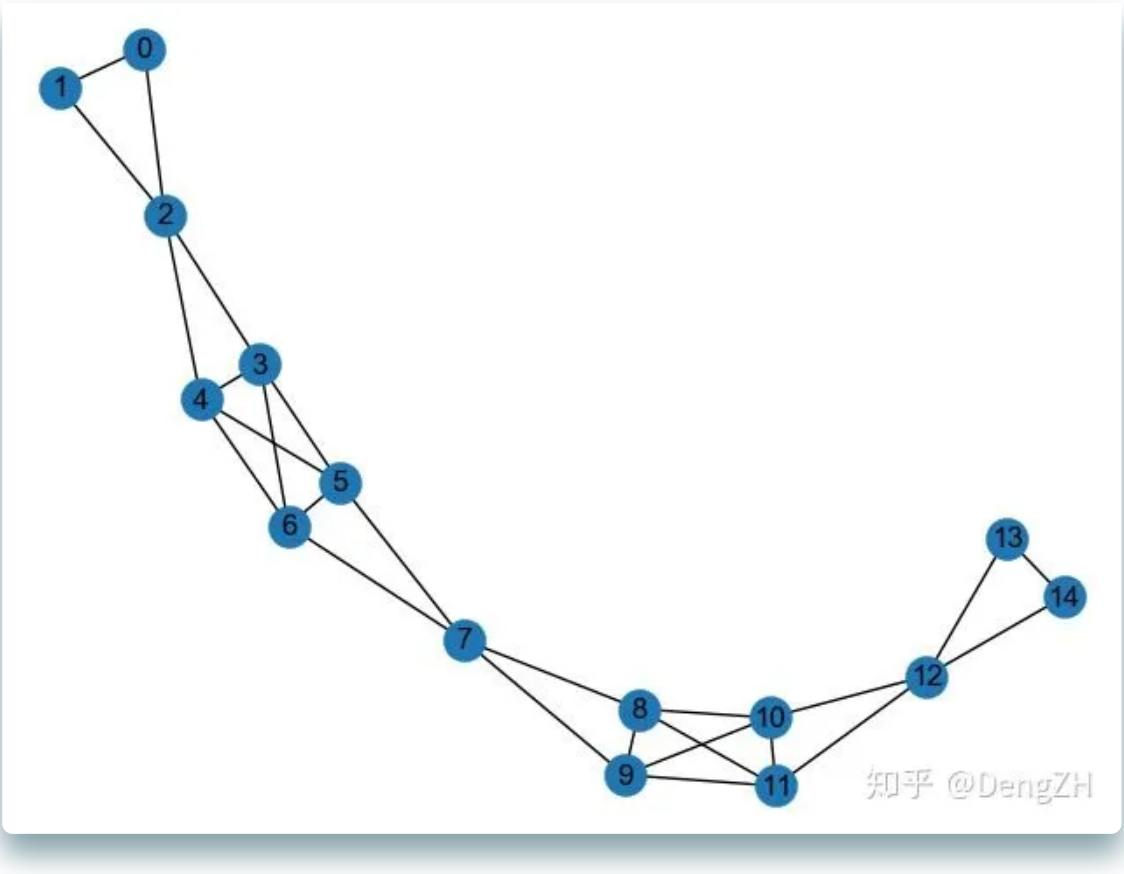
更好地感知结点所处的局部结构。

接下来，尝试用自己构造的网络来实验，看看结果是否会和上述分析一致。为了能观察到期望的结果，构造的网络必须：

- 有一定的聚簇现象；
- 包含密集连接的局部结构。

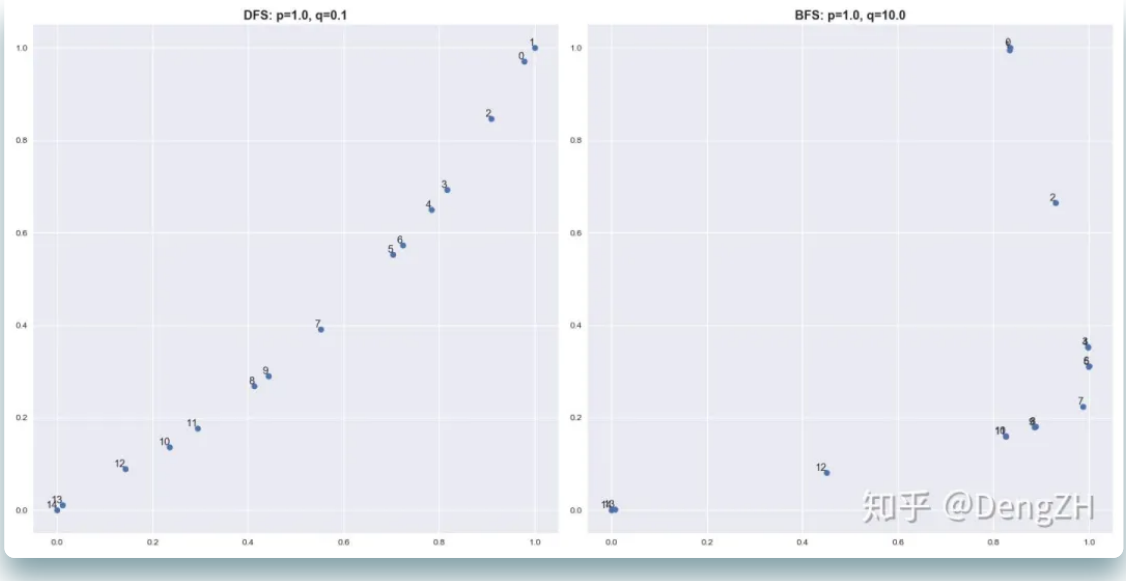
构造的网络比较小，embed 的维数可以直接设置为2，这样可以直接 plot 在 2D 平面上直观地通过距离来衡量结点之间的相似度。随机游走序列的长度设置为10。

首先，测试一下这个像 bridge 一样的网络：



img

这个网络是对称的，有一个中心点 7，左右各有一个三角形局部结构和四边形的密集连接结构。将 node2vec 学到的结点 embedding 画出来：

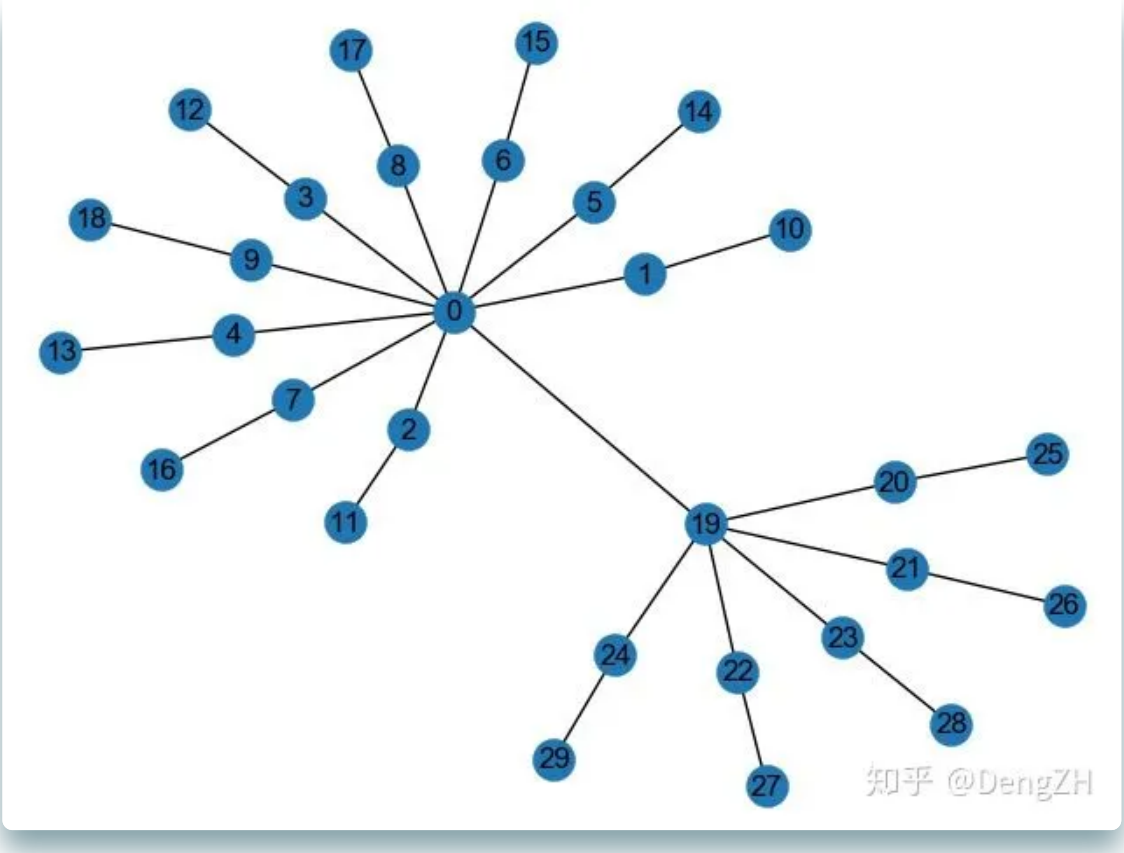


img

我们首先分析一下局部结构，可以看到 BFS 对局部结构非常敏感，同处一个局部结构内的结点的 embedding 几乎相同（比如结点0和1），这与之前的分析一致。另外，我们也可以观察到，比起 DFS，BFS 得到的 embedding 还有个特点，局部结构内的结点跟以外的结点有着明显的划分，即使是相邻结点也可能得到很不一样的embedding，比如：结点12 是连接两个局部结构的点，在 DFS 中它与相邻的结点10跟11距离较短，而在 BFS 中则相距较远。这个观察其实在一定程度上体现出了 Figure 3 中对结构性的诠释，但是我们也可以看到，处于对称结构另一侧的结点2和结点12同样相距较远，并没能得到像 Figure 3 那么漂亮的结果。

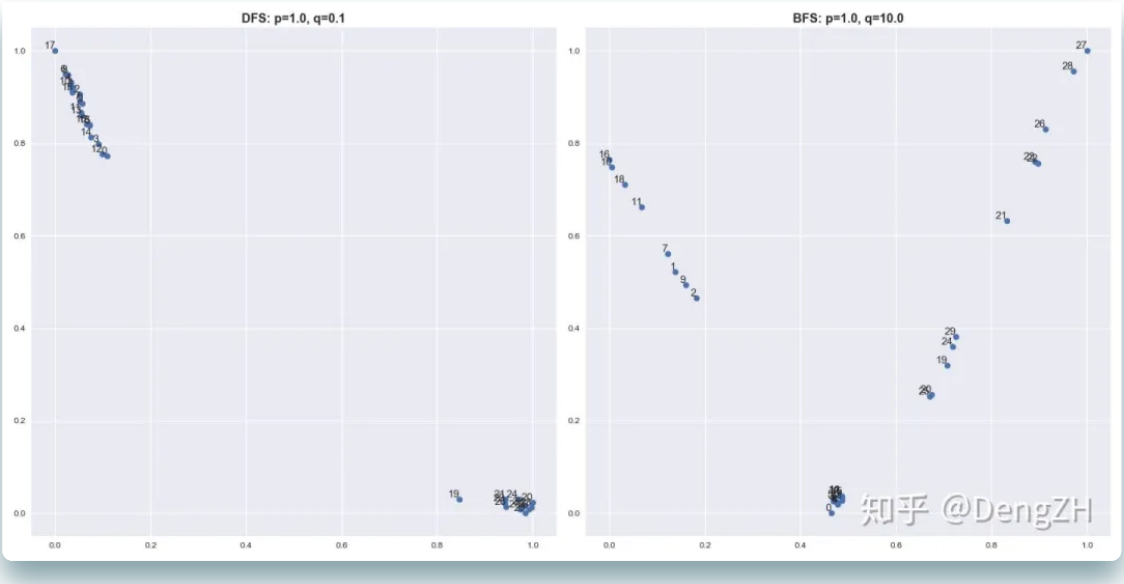
然后再分析一下聚类效果。可以看到 DFS 得到的 embedding 分布比较均匀，不像 BFS 那样会出现比较大的差距。设定的游走序列长度为10，窗口大小为5，在这个网络中相邻结点有类似的上下文的可能性是比较高的，比方说结点10、11和结点12，它们的上下文会比较像，因此 embedding 的结果也会比较像。在 BFS 中上下文不会那么相似，因此 embedding 结果也就会差距更大一些。聚类的话，其实这个网络聚簇现象并不明显，所以 DFS 的结果没有看出有很明显的聚类边界，更倾向于把整个网络分为一个簇；而 BFS 就很明显地把这个网络分为了5个簇，密集连接的部分分到一个簇中，两边的边缘结点各一个簇，两个作为连接枢纽的结点各一个簇。

再测试一下这个像花一样的网络：



img

这个网络包含两朵分别以结点 0 和结点 19 为中心的花。将 node2vec 学到的结点 embedding 画出来：



img

可以看到 DFS 在这个网络中很好地学习到了两个簇（两朵花）的边界，把同一簇的结点 embedding 推到一起，把不同簇之间的距离尽可能拉开。而 BFS 得到的聚类结果就比较糟糕了，可以看到分属两朵花的结点在 embedding 空间中还是有所区分的，但由于 BFS 对局部结构非常敏感，所以在学习 embedding 的拉扯过程中，两个相邻的中心结点 0 和 19 之间的距离无法被推远，这也使得别的结点的 embedding 学习收到影响，无法像 DFS 中那样分散到两个不同的簇中。但是，我

们也可以注意到，这种情况下，一些处于边界区域的结点有可能会聚类到一起，比如结点 2 跟结点 29 之间的距离要小于结点 2 跟结点 0 之间的距离。这也在某种程度上体现出了 Figure 3 中表达的结构性，但无法得到那么完美的图像。

通过在这两个网络上进行实验，基本上验证了前面分析的正确性。虽然实验中没能复现出像论文 Figure 3 那么完美的结果，但这并没有否定 node2vec 的效果。一方面，现实中的网络数据不会像这两个网络这么简单，实现不同的任务，使用不同的数据需要的 p 和 q 也不一样，未必要像上面的设置这么极端。另一方面，实际任务中要求的 embedding 不会只要同质性或者只要结构性，一般都是两者兼备。真实数据一般还会包含结点属性和边的属性，这些属性数据对 embedding 的学习也是至关重要的。

感兴趣的同学可以下载代码自己尝试一下，欢迎交流

https://github.com/familyld/A_Simple_node2vec_Example

更多关于图神经网络/图表示学习/推荐系统, 欢迎关注我的公众号 【图与推荐】

阅读原文