加速属性网络嵌入

研读笔记

中南大学彭汪祺 2021/6/29

1. 背景

在现实世界中，存在大量复杂网络，这些网络往往可以表征为一个个高维向量空间，这些高维空间中隐藏着许多有效特征。但由于一个信息网络动辄可能包含数十亿个节点和边，因此在整个网络上执行复杂的推理过程，可能会非常棘手，需要人们通过大量工程试验和人工努力去提取和识别，例如节点分类、链接预测、网络聚类等等。对于这个问题，人们提出了网络嵌入算法，**将高维且稀疏的向量空间映射成低维且稠密的向量空间，既可以很好地保持原始网络的邻近性，也较大程度上减轻了提取网络特征的计算难度**。

现有的网络嵌入算法集中于纯粹的网络拓扑结构上，却忽视了节点属性网络的重要性。事实上，现实网络中的节点不仅存在拓扑关系，而且通常有着一组丰富的属性，社会科学理论证明**节点属性与网络结构是高度相关的，并表明如果将网络结构与节点属性进行共同分析，将会提高网络嵌入算法的学习性能。因此，研究节点属性对于网络嵌入的影响非常有必要**。

属性网络嵌入同样也要面临大规模且无序网络的挑战，主要存在以下三个挑战：

(1) 减小算法运行的时间复杂度；
(2) 使嵌入算法具有可伸缩性，从而更好地评估节点邻近性；
(3) 网络结构和节点属性这两大来源都会存在噪声和不完整数据，需要增强鲁棒性。

2. 论文工作

针对上述提到的构建网络结构和节点属性联合空间，并在空间中进行有效节点邻近性分析，同时提高模型学习效率和可扩展性的问题，论文做了以下工作：

(1) 从形式上定义了属性网络嵌入的问题；
(2) 提出了一个可扩展的、效率高的网络嵌入框架 AANE；
(3) 提出了一种分布式优化算法，使 AANE 处理节点更为高效；
(4) 在真实数据集上验证了 AANE 的有效性和效率。

3. 论文模型建立

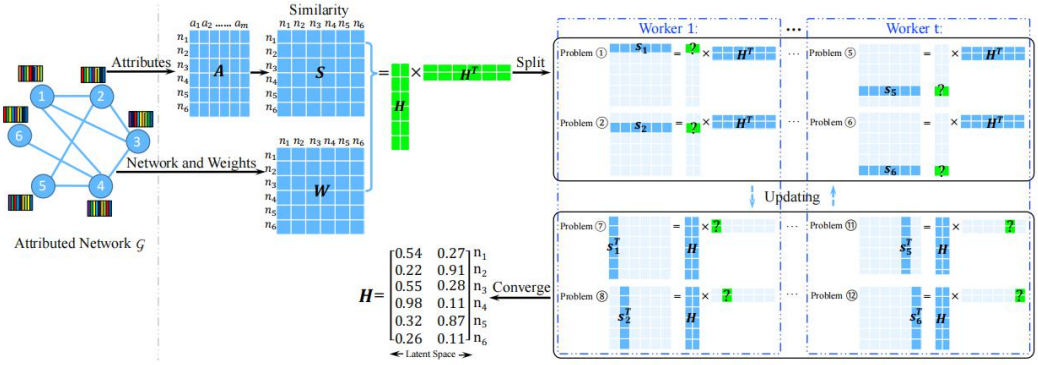
3.1 AANE 的基本介绍

论文提出的 AANE 框架能够满足一个理想的嵌入算法应该具有的三大要求：

(1) 它可以处理任意类型的网络，无论该网络是否有向，是否存在边权重；
(2) 无论是在结构网络还是属性网络，都可以保持节点的邻近性；

(3) 具有较好的伸缩性，可以应对大规模网络。

AANE 之所以具有高效性，是因为它创造性地将嵌入工作同时交给多个 workers 独立、同时间完成，大大地节省了时间。AANE 主要分为两大部分，即结构网络与属性网络，这两个网络具有高度的依赖性和相关性。这里设网络的节点数位 n，每个节点包含 m 种属性，n × n 矩阵 W 为结构网络矩阵，矩阵 W 中记录着网络的拓扑结构以及各边的权重，n × m 矩阵 A 为属性网络矩阵，它存储着 n 个节点的 m 组属性，属性网络矩阵 A 可以计算得到 n × n 矩阵 S，S 代表属性相似度，最后原始网络可以降维成矩阵 H。模型的建立过程可以见下图：



首先从原始网络中得出属性网络 A 和结构网络 W，这里 A 计算得到节点相似度矩阵 S，S 可以拆分为 H 和H^T的点积，W 同样发挥了作用——这里在分解过程中引入基于边的惩罚，这样将使得相似节点在矩阵 H 上更加靠近。**AANE 采用了一种分布式算法来加快优化速度，即将原问题分解成 2*n 个低复杂度的子问题，前 n 个子问题互相独立，后 n 个同样如此，所有的子问题可以分配给 t 个工作者来同时完成，从而大大节省了网络嵌入算法的学习时间。从该示例网络中，我们可以看到，在最终输出的 H 矩阵中，节点 1 和 3 由非常相似的向量表示，从网络中也可以发现它们二者在结构和属性上具有高度相似性，从而印证了模型得到的结论。**

3.2 结构网络模型

AANE 在结构网络和属性网络中都应该保持良好的节点邻近性。这里在结构网络中，设h_i和h_j是节点 i 和 j 的向量表示，而w_{ij}是节点 i 和 j 之间的边权重，因此研究者提出以下损失函数来最小化节点之间的嵌入差异：

$$\mathcal{J}_{\mathcal{G}} = \sum_{(i,j)\in\mathcal{E}} w_{ij}\|\mathbf{h}_i - \mathbf{h}_j\|_2$$

该损失函数中，为了能够将损失函数降到最低，i-j 边权重w_{ij}越大，则h_i和h_j之间的差距就必须越小。而采用ℓ_2范数的好处在于，可以减轻异常值和缺失数据带来的影响，增强模型的鲁棒性。

3.3 属性邻近性模型

研究者认为，结构网络和属性网络具有高度依赖性和相关性，因此结构网络中节点邻近度与属性网络中节点的邻近度应该一致。研究者使h_i和h_j的点积与对应属性相似性s_{ij}相同，将属性相似性矩阵 S 通过对称分解，分解为 H 和H^T。

因此，将损失函数定义为：

$$\mathcal{J}_A = \left\|\mathbf{S} - \mathbf{H}\mathbf{H}^\top\right\|_{\mathrm{F}}^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\mathbf{s}_{ij} - \mathbf{h}_i\mathbf{h}_j^\top\right)^2$$

3.4 模型的联合表示

得到两个网络的损失函数后，为了使它们往统一的鲁棒性和信息空间的方向进行互补，将两个损失函数结合为一个统一的损失函数如下：

$$\min_{\mathbf{H}} \mathcal{J} = \left\|\mathbf{S} - \mathbf{H}\mathbf{H}^{\top}\right\|_{\mathrm{F}}^{2} + \lambda \sum_{(i,j)\in\mathcal{E}} w_{ij}\|\mathbf{h}_i - \mathbf{h}_j\|_2$$

其中，λ 是作为结构网络和属性网络之间的平衡参数出现的，同时作为一个正则化参数，来平衡簇数。

3.5 加速算法和分布式算法

这一部分是模型的关键部分，它实现了模型的高效分布式算法，大大提高了模型的运行效率。上面提出的模型联合表示中，7相对于h_i来说是可分离的，那样就可以将问题重写为一个双凸优化问题。具体的推导过程如下：

首先设 Z=H，那样模型的联合表示可改为：

$$\min_{\mathbf{H}} \sum_{i=1}^{n} \left\|\mathbf{s}_i - \mathbf{h}_i\mathbf{Z}^{\top}\right\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} w_{ij}\|\mathbf{h}_i - \mathbf{z}_j\|_2$$

这里7相对于h_i和z_i来说是可分离的，又由于ℓ_2范数是凸的，故而上式是一个双凸型的，也就是说，当z_i固定时，w，r，t 和h_i都是凸的，同理h_i固定时，w，r，t 和z_i都是凸的。这样一来，我们可以将原始的复杂问题分解成 2*n 个小的凸优化子问题。

具体分解的过程，研究者借鉴了 ADMM 算法，一种分布式凸优化技术，将整个复杂问题通过 2*n 个更新步骤来加速优化，这种方式有三个优点：

(1) 前 n 个和后 n 个更新步骤都是相互独立的，每次迭代可以分配给工作人员，而不需要一个固定的顺序；

(2) 更新步骤复杂度低；

(3) 快速收敛。

因此，研究人员提出了一个增广拉格朗日式子如下：

$$\begin{aligned}\mathcal{L} = &\sum_{i=1}^{n} \left\|\mathbf{s}_i - \mathbf{h}_i\mathbf{Z}^{\top}\right\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} w_{ij}\|\mathbf{h}_i - \mathbf{z}_j\|_2 \\ &+ \frac{\rho}{2}\sum_{i=1}^{n}\left(\|\mathbf{h}_i - \mathbf{z}_i + \mathbf{u}_i\|_2^2 - \|\mathbf{u}_i\|_2^2\right)\end{aligned}$$

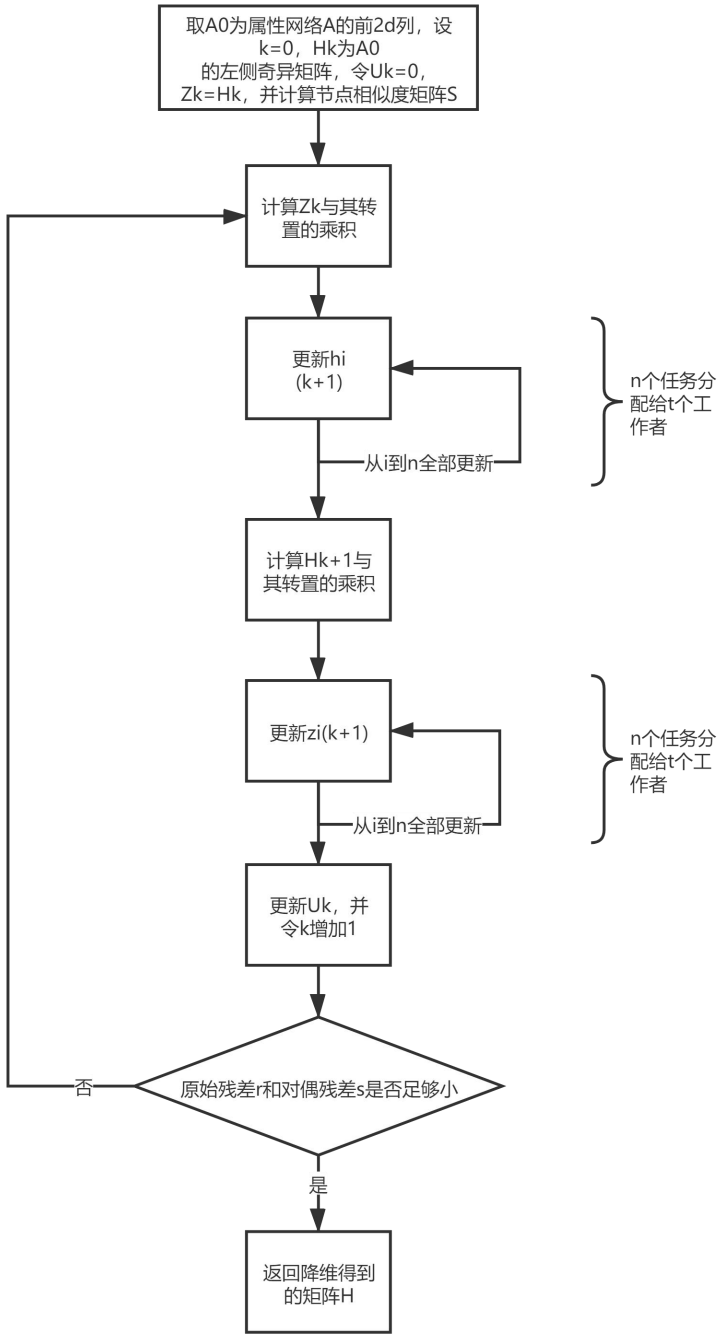
其中u_i是缩放的对偶变量，而且 ρ 作为惩罚参量。通过设定变量偏导数为 0 的方式，得到以下更新规则：

$$\mathbf{h}_i^{k+1} = \frac{2\mathbf{s}_i\mathbf{Z}^{k} + \lambda\sum_{j\in N(i)} \frac{w_{ij}\mathbf{z}_j^{k}}{\left\|\mathbf{h}_i^{k} - \mathbf{z}_j^{k}\right\|_2} + \rho\left(\mathbf{z}_i^{k} - \mathbf{u}_i^{k}\right)}{2\mathbf{Z}^{k}\mathbf{Z}^{k} + \left(\lambda\sum_{j\in N(i)} \frac{w_{ij}}{\left\|\mathbf{h}_i^{k} - \mathbf{z}_j^{k}\right\|_2} + \rho\right)\mathbf{I}}$$

$$\mathbf{z}_i^{k+1} = \frac{2\mathbf{s}_i\mathbf{H}^{k+1} + \lambda\sum_{j\in N(i)} \frac{w_{ij}\mathbf{h}_j^{k+1}}{\left\|\mathbf{z}_i^{k} - \mathbf{h}_j^{k+1}\right\|_2} + \rho\left(\mathbf{h}_i^{k+1} + \mathbf{u}_i^{k}\right)}{2\mathbf{H}^{k+1}\mathbf{H}^{k+1} + \left(\lambda\sum_{j\in N(i)} \frac{w_{ij}}{\left\|\mathbf{z}_i^{k} - \mathbf{h}_j^{k+1}\right\|_2} + \rho\right)\mathbf{I}}$$

对于h_i^k来说，已经有前人的工作证明了它的导数呈现单调递减性质，同时它又是一个凸函数，因此必然存在$h_i^k = h_i^{k+1}$，使得它为最大值。迭代最开始，我们设A_0为 A 的前 2d 列，H 为A_0的左奇异向量。研究者的优化策略是将复杂问题分解为 2*n 个子问题，并逐个迭代地

去解决它们，每次迭代的 n 个更新步骤，可以分布式地分配给 t 个工作者，直至原始残差 r 和对偶残差 s 足够小。具体的迭代步骤如下所示：



4.实验

提出 AANE 模型后，研究人员从性能和效率两个方面，将 AANE 与其他降维模型进行了比较。

实验采用的数据集来自真实世界中的三个网络社区，分别为 BlogCatalog、Flicker、Yelp[2]。这些社区数据集都有一个共同点——不仅存在用户之间的各类关系，每个用户还拥有自己的标签属性，这为 AANE 的属性网络搭建提供了基础。数据集的具体情况如下表所示：

Dataset	BlogCatalog	Flickr	Yelp
Nodes ($\|\mathcal{V}\|$)	5,196	7,564	249,012
Edges ($\|\mathcal{E}\|$)	171,743	239,365	1,779,803
Attribute (m)	8,189	12,047	20,000
Label (ℓ)	6	9	11

4.1 评估 AANE 的性能

AANE 模型中加入了节点属性网络，与结构网络一起作为模型的来源。这里为了评估节点属性对于网络嵌入的贡献，将 AANE 模型与 DeepWalk、LINE、PCA 等常用的降维技术，以及 LCMF、MultiSpec 这类先进的属性网络学习方法放在一起进行比较。

评估实验中，需要完成的任务为基于网络的训练，创建分类器，来预测新节点属于哪个或哪些类别。首先将整个节点集合随机分成一个训练组和一个测试组，并去除训练组合测试组节点之间的所有边。这里的数据集存在多个标签类别，研究者基于每个类别构建一个二进制 SVM 分类器，网络嵌入最终降维得到的矩阵维度 d=100，所有试验结果为 10 次运行后得到的算术平均值。

同时，研究者还将训练节点的比例调整为 10%、25%、50%等，评估不同大小的训练数据带来的效果。

分类性能基于宏观平均和微观平均两个指标来评价，指标越大，说明性能越优秀。得到的测试结果如下所示：

		BlogCatalog				Flickr				Yelp 1			
Training Percentage		10%	25%	50%	100%	10%	25%	50%	100%	10%	25%	50%	100%
# nodes for embedding		1,455	2,079	3,118	5,196	2,118	3,026	4,538	7,564	13,945	19,921	29,881	49,802
Macro-average	DeepWalk	0.489	0.548	0.606	0.665	0.310	0.371	0.462	0.530	0.139	0.159	0.215	0.275
	LINE	0.425	0.542	0.620	0.681	0.256	0.331	0.418	0.512	0.165	0.173	0.193	0.227
	PCA	0.691	0.780	0.821	0.855	0.510	0.612	0.671	0.696	0.591	0.599	0.605	N.A.
	LCMF	0.776	0.847	0.886	0.900	0.585	0.683	0.729	0.751	0.589	0.605	0.612	N.A.
	MultiSpec	0.677	0.787	0.847	0.895	0.589	0.722	0.802	0.859	0.461	0.460	N.A.	N.A.
	AANE	**0.836**	**0.875**	**0.912**	**0.930**	**0.743**	**0.814**	**0.852**	**0.883**	**0.630**	**0.645**	**0.656**	**0.663**
Micro-average	DeepWalk	0.491	0.551	0.611	0.672	0.312	0.373	0.465	0.535	0.302	0.310	0.318	0.350
	LINE	0.433	0.545	0.624	0.684	0.259	0.332	0.421	0.516	0.230	0.243	0.264	0.294
	PCA	0.695	0.782	0.823	0.857	0.508	0.606	0.666	0.692	0.667	0.674	0.681	N.A.
	LCMF	0.778	0.849	0.888	0.902	0.576	0.676	0.725	0.749	0.668	0.680	0.686	N.A.
	MultiSpec	0.678	0.788	0.849	0.896	0.589	0.720	0.800	0.859	0.553	0.571	N.A.	N.A.
	AANE	**0.841**	**0.878**	**0.913**	**0.932**	**0.740**	**0.811**	**0.854**	**0.885**	**0.679**	**0.694**	**0.703**	**0.711**

同时，因为三种基于属性网络的模型在 Yelp 数据集中不可行，因此研究者从中随机选择 20%作为新数据集，得到以下测试结果：

Training Percentage		10%	25%	50%	100%
# nodes for embedding		69,723	99,605	149,407	249,012
Macro-average	DeepWalk	0.239	0.254	0.266	0.260
	LINE	0.216	0.236	0.259	0.279
	AANE	**0.649**	**0.659**	**0.660**	**0.665**
Micro-average	DeepWalk	0.324	0.345	0.366	0.368
	LINE	0.295	0.313	0.336	0.354
	AANE	**0.698**	**0.709**	**0.711**	**0.714**

从测试结果可以看：

(1) AANE 模型在所有数据集的所有训练比例中，都得到了最优的性能；

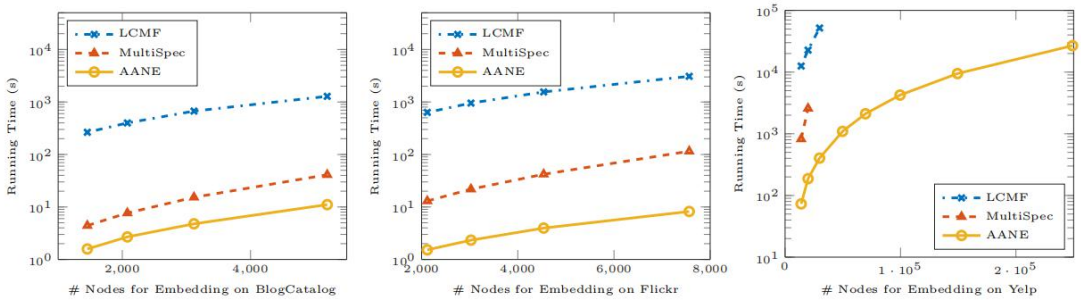
(2) 基于属性网络的模型 LCMF、MultiSpec 和 AANE 都优于基于结构网络的模型；

(3) 随着训练节点的比例增加，训练得到的性能也在增加。

总体来说，AANE 在性能方面表现优秀，优于其它所有方法。

4.2 评价 AANE 的效率

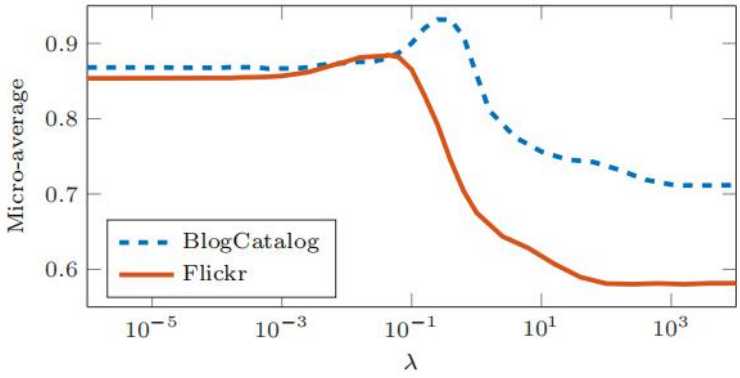
研究者选择 LCMF、MultiSpec 和 AANE 进行效率比较，同样运行在三个数据集上，得到每个数据集上运行所需的时间如下：



很明显，除了 LCMF 和 MultiSpec 无法在 Yelp 上运行，并进行比较外，AANE 的运行时间都是最低的，同时，随着节点数量（网络规模）的增大，所需运行的时间也会相应增加。这有力地说明了 AANE 模型采用分布式算法带来的高效性和可伸缩性。

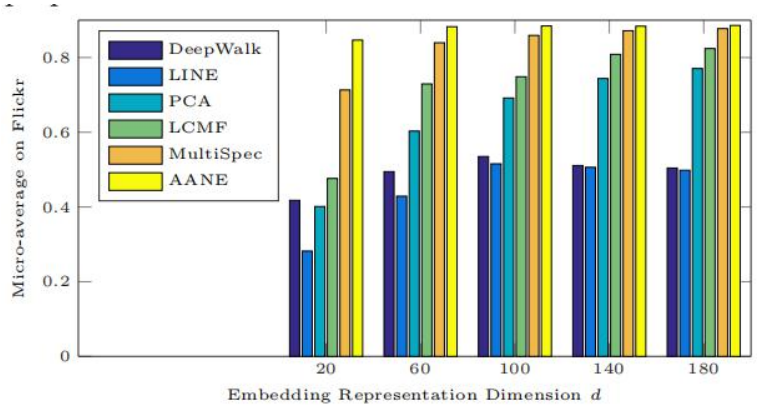
4.3 评估参数对模型的影响

模型中采用了两个参数 λ 和 d， λ 是作为结构网络和属性网络之间的平衡参数出现的，而 d 是训练得到矩阵 H 的维数。这里研究者对 λ 对 AANE 模型的影响做了研究，通过改变 λ 的值，探究 AANE 在 BlogCatalog 数据集和 Flicker 数据集上的不同表现，具体如下图所示：



从图中可以看出，当 λ 小于 0.1 时， λ 的变化对 AANE 模型的测试结果基本没有影响，当 λ 继续增大时，AANE 模型受到拓扑结构的影响也增大，模型的准确度开始出现明显下降。

维数 d 对于所有的模型的影响如下：



从图中可以看出，d 不断增大时，所有模型的测试性能总体上是上升的，但对 AANE 的模型测试结果来说，变化很小，并且 AANE 模型的性能始终优于其它模型。这说明要求的嵌入维度较低时，相对于其它模型，AANE 仍然能够保持较好的性能。

5.结论

这篇论文工作中最大的创新点在于采用分布式的计算方法，将一个复杂的问题分解为多个独立的子问题，分配给多个工作者同时进行，大大提高了网络嵌入模型的运行速度；其次，论文注意到了节点属性在网络嵌入中的重要作用，提出了拓扑结构-节点属性联合嵌入模型，提高了模型学习低维表示的准确度。

从模型的各类测试结果中看，AANE 模型取得了非常优秀的结果，无论是低维学习的准确度，还是模型运行的效率，都优于测试中的所有其它模型，具有很高的性能和效率。

6. 模型优化建议

关于模型的优化，在论文最后，作者亲自提出了两个重要的发展方向：

1. 如何嵌入具有动态拓扑结构和节点属性的大规模属性网络？
2. 如何将 AANE 扩展到半监督框架？

同时，通过我自己对其它有关网络嵌入文献的阅读了解，我认为模型还可以向以下的方向进行一些改进：

（1）考虑边缘属性的存在。论文中联合了拓扑结构和节点属性进行学习，但是我认为真实世界中的网络，不仅节点存在属性，网络的边也存在属性，例如一个社交网络中，不同用户之间的关系是不同的，例如父子、夫妇、兄弟姐妹、朋友、同事等关系，**这些不同关系都属于社交网络边缘存在的属性，这些边缘属性是否也能够利用起来，和现有的拓扑结构、节点属性联合进行模型学习呢？**

（2）考虑异构网络的存在。在一个网络中，并非所有边和节点的性质都是一样的，具有多类节点或边缘的网络称为异构网络，异构网络中的拓扑结构、节点属性等，必然要比普通的、只有同种类型的节点边缘的网络更加复杂，如何在异构网络中提出改进的 AANE 数学模型呢？