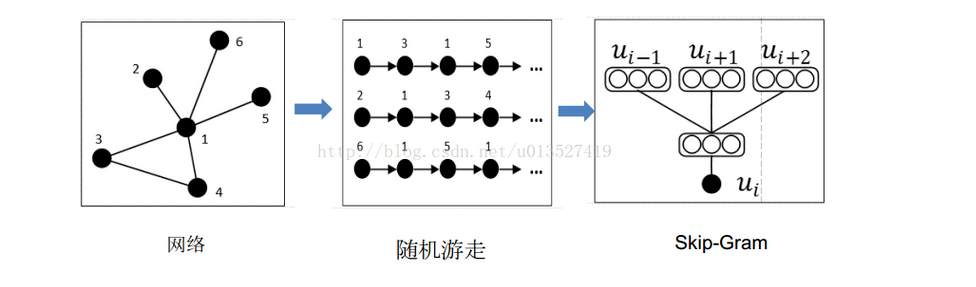
Deep walk

图表示学习baseline

主要思想：不用多说，算是表示学习的起点，借鉴自然语言处理，主要通过

1. random walk 获得网络节点随机游走序列，来描述网络结构
2. word2vec 通过word2vec学习嵌入向量，与NPL一模一样

特点：随机游走是无偏的，这种游走生成的节点序列，更加接近图的深度优先搜索，偏向于探寻网络节点高阶的距离较远的空间拓扑结构，个人觉得，这种随机游走方式对于领域和高阶的网络结构探索都存在一定的不确定性，也不够充分。

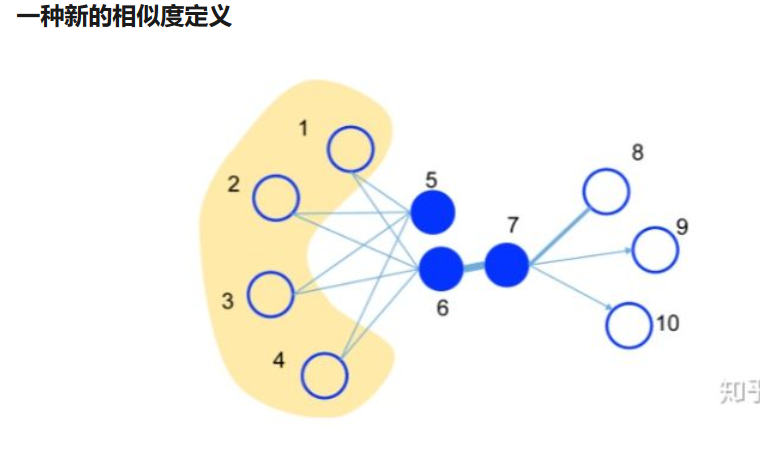


Line（Large-scale Information Network Embedding）

主要思想：line算法更加关注领域网络结构，更加类似于广度优先搜索，学到的网络结构更加接近局部结构

Line最大的不同在于 它算是有监督的学习，标签就是边和权重，利用顶点之间向量内积映射的分布去拟合存在的边权重分布

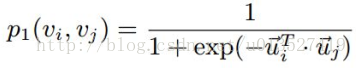
LINE还可以应用在带权图中(DeepWalk仅能用于无权图)。



一阶相似度

1阶相似度用于描述图中成对顶点之间的局部相似度，形式化描述为若 u, v 之间存在直连边，则边权Wuv 即为两个顶点的相似度，若不存在直连边，则1阶相似度为0。 如上图，6和7之间存在直连边，且边权较大，则认为两者相似且1阶相似度较高，而5和6之间不存在直连边，则两者间1阶相似度为0。

通过已经得到的一阶相似度，去学习节点的嵌入向量



v代表节点，u代表节点的embedding。上面式子的意思是两节点越相似，內积越大，sigmoid映射后的值越大，

同时定义经验分布

优化目标为最小化：

d是两个分布的距离，常用的衡量两个概率分布差异的指标为KL散度，K-L散度其实是数据的原始分布p和近似分布q之间的对数差值的期望。

使用KL散度并忽略常数项后有



总结起来大概就是说，用各个节点的嵌入向量的内积sigmoid映射分布去拟合真正的各个边的权重分布，目标函数就是两个分布之间的散度，最小化散度，这样训练得到的嵌入向量学习了边权重的信息，将边的结构和权重转化为个节点的嵌入向量来表示。

在我们已经知道了边的权重分布的情况下，用每一个节点的边和权重构成经验分布函数，嵌入向量的表示尽可能拟合这个分布,这里一阶相似度中，两个分布都是的基本单位是全图中任意一对顶点

用整个图去训练，用任何一对顶点的内积去拟合两点之间的权重，用整体的内积分布拟合整体的权重

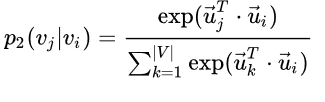
对于某一无权图来说，节点100个，那么一阶相似度函数的最终结果应该有100\*100个

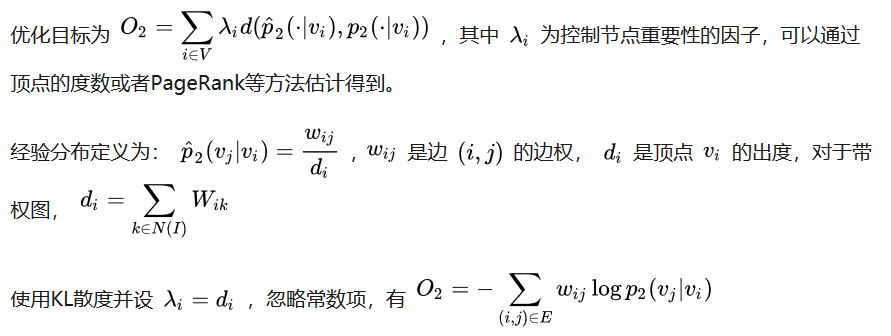
边的经验分布函数结果也有100\*100，只不过有的位置是0有的位置是1，如果是有权图，那这个位置就是两个节点之间的权值

二阶相似度

仅有1阶相似度就够了吗？显然不够，如上图，虽然5和6之间不存在直连边，但是他们有很多相同的邻居顶点(1,2,3,4)，这其实也可以表明5和6是相似的，而2阶相似度就是用来描述这种关系的。 这里对于每个顶点维护两个embedding向量，一个是该顶点本身的表示向量，一个是该点作为其他顶点的上下文顶点时的表示向量。

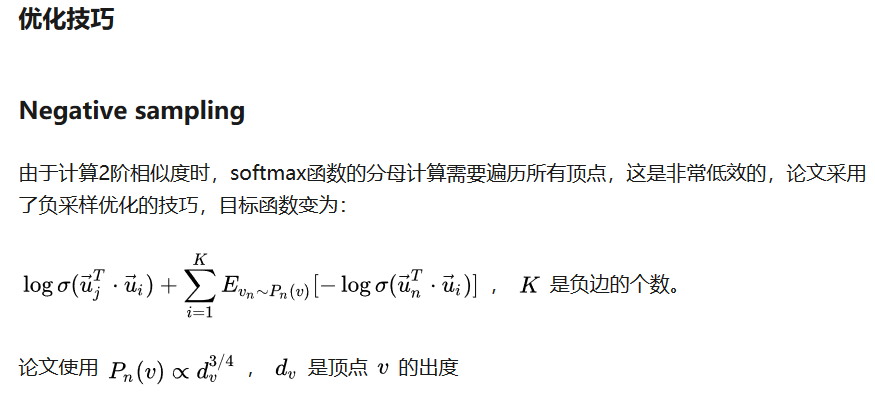
对于有向边(i, j)，定义给定顶点 Vi条件下，产生上下文(邻居)顶点 Vj的概率为





不同之处就是，这次两个分布都是条件概率分布，同样用KL散度评估两个分布差异。

那嵌入向量是如何真正的学到了二阶的相似度的呢，例如上图中5,6两个顶点二阶相似度应该是比较高的，那么他们的嵌入向量应该非常接近，怎么得到这个结果呢，其实二阶相似度要学习的目标是5,6这种经验分布函数非常相似的节点结构，5,6的经验分布函数是一模一样的，5和6两个顶点的嵌入向量在拟合经验分布函数的时候，最终达到学习目的。最终的结果，二阶上类似5,6这种结构其嵌入向量应该是非常接近的。



没看懂这目标函数

## Edge Sampling

另一种方法则是从原始的带权边中进行采样，每条边被采样的概率正比于原始图中边的权重，这样既解决了学习率的问题，又没有带来过多的存储开销。这里的采样算法使用的是**Alias**算法，Alias是一种 时间复杂度的离散事件抽样算法

## 低度数顶点

对于一些顶点由于其邻接点非常少会导致embedding向量的学习不充分，论文提到可以利用邻居的邻居构造样本进行学习，这里也暴露出LINE方法仅考虑一阶和二阶相似性，对高阶信息的利用不足。

## 模型和损失函数定义

LINE使用梯度下降的方法进行优化，直接使用tensorflow进行实现，就可以不用人工写参数更新的逻辑了~

这里的 实现中把1阶和2阶的方法融合到一起了，可以通过超参数order控制是分开优化还是联合优化，论文推荐分开优化

首先输入就是两个顶点的编号，然后分别拿到各自对应的embedding向量，最后输出内积的结果。 真实label定义为1或者-1，通过模型输出的内积和line\_loss就可以优化使用了负采样技巧的目标函数了

## nodo2vec

## <http://www.sohu.com/a/124091440_355140>

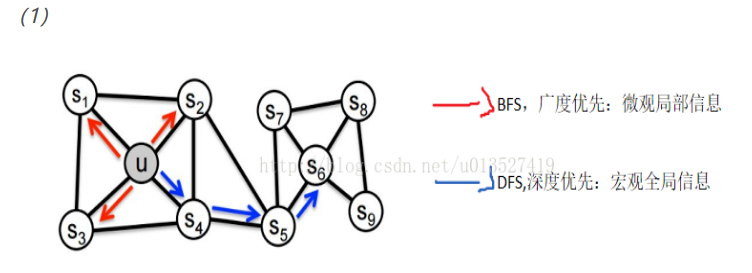
## 微信lookalike应用

1 随机游走策略变化

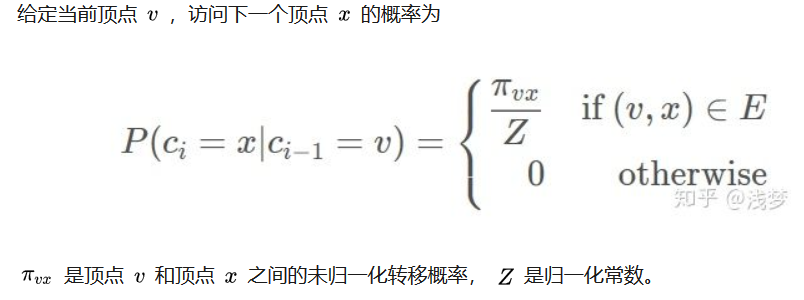
通过调整随机游走权重的方法使graph embedding的结果在网络的同质性（homophily）和结构性（structural equivalence）中进行权衡权衡。

具体来讲，网络的“结构性”指的是距离相近节点的embedding应该尽量近似，如图4，节点u与其相连的节点s1、s2、s3、s4的embedding表达应该是接近的，这就是“同质性“的体现。

“同质性”指的是结构上相似的节点的embedding应该尽量接近，图4中节点u和节点s6都是各自局域网络的中心节点，结构上相似，其embedding的表达也应该近似，这是“结构性”的体现。



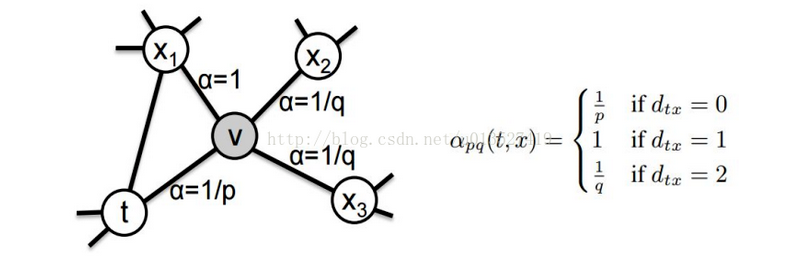
node2vec依然采用随机游走的方式获取顶点的近邻序列，不同的是node2vec采用的是一种有偏的随机游走，



node2vec引入两个超参数 p和 q来控制随机游走的策略，

参数控制跳转概率的随机游走，之前完全随机时，p=q=1.

--返回概率参数（Return parameter）p，对应BFS，p控制回到原来节点的概率，如图中从t跳到v以后，有1/p的概率在节点v处再跳回到t。

--离开概率参数（In outparameter）q，对应DFS，q控制跳到其他节点的概率。

2目标函数重新定义

采样完顶点序列后，剩下的步骤就和deepwalk一样了，用word2vec去学习顶点的embedding向量。 值得注意的是node2vecWalk中不再是随机抽取邻接点，而是按概率抽取，node2vec采用了Alias算法进行顶点采样。

这里没看太懂，后面又说，节点序列采样完成之后，就和deepwalk一样了，那没什么可说的了

node2vec所体现的网络的同质性和结构性在推荐系统中也是可以被很直观的解释的。同质性相同的物品很可能是同品类、同属性、或者经常被一同购买的物品，而结构性相同的物品则是各品类的爆款、各品类的最佳凑单商品等拥有类似趋势或者结构性属性的物品。毫无疑问，二者在推荐系统中都是非常重要的特征表达。由于node2vec的这种灵活性，以及发掘不同特征的能力，甚至可以把不同node2vec生成的embedding融合共同输入后续深度学习网络，以保留物品的不同特征信息。

## 4 SDNE

﻿SDNE（Structural Deep Network Embedding ）是和node2vec并列的工作，均发表在2016年的KDD会议中。可以看作是基于LINE的扩展，同时也是第一个将深度学习应用于网络表示学习中的方法。SDNE使用一个自动编码器结构来**同时**优化1阶和2阶相似度(LINE是分别优化的)，学习得到的向量表示能够保留局部和全局结构，并且对稀疏网络具有鲁棒性。

## 5 Struct\_vec

