使用老师们实现的GraphSGAN

学习过程参考以下博客：

## 1. Graph representation learning

(network embedding / graph embedding / network representation learning) tries to embed each node of a graph into a low-dimensional vector space, which preserves the structural similarities or distances among the nodes in the original graph.

## 2. Graph分类：

****按照Input：****

* Homogeneous graph (e.g., citation network)
* Heterogeneous graph

             Multimedia network

             Knowledge graph (entity,relation)

* Graph with side information（辅助信息）

       Node/edge label (categorical)  
       Node/edge attribute (discrete or continuous)  
       Node feature (e.g., texts)

* Graph transformed from non-relational data (从非关系型数据中转换成的图)

         Manifold learning

****按照Output:****

* Node embedding (the most common case)
* Edge embedding

        Relations in knowledge graph  
       Link prediction

* Sub-graph embedding

      Substructure embedding  
      Community embedding

* Whole-graph embedding

          Multiple small graphs, e.g., molecule, protein  
****按照Method:****

* Matrix factorization

    Singular value decomposition  
    Spectral decomposition (eigen-decomposition)

* Random walk
* Deep learning

    Auto-encoder(SDNE)

   Convolutional neural network

* Self-defined loss (LINE)

    Maximizing edge reconstruction probability  
    Minimizing distance-based loss  
    Minimizing margin-based ranking loss

## 3.  Motivation

 网络表示学习方法可以分成两个类别。

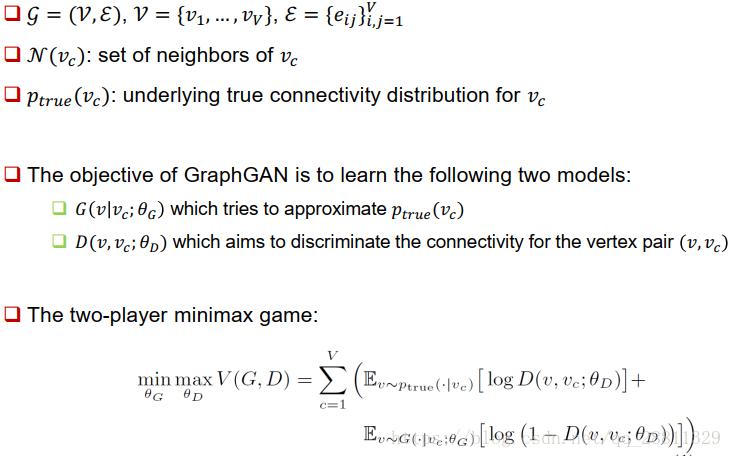
一种是Generative model（生成式模型），假定对于每一个顶点，在图中存在一个潜在的、真实的连续性分布 Ptrue(v|vc)， 图中的每条边都可以看作是从Ptrue里采样的一些样本。生成式方法都试图将边的似然概率最大化，来学习vertex embedding。例如DeepWalk (KDD 2014) and node2vec (KDD 2016)。

Discriminative Model（判别式模型）将两顶点联合作为feature，预测两点之间存在边的概率。例如SDNE (KDD 2016) and PPNE (DASFAA, 2017)。

LINE (WWW 2015) 尝试将两者结合起来。而最近非常popular的GAN设计了一个 game-theoretical minimax game 将两者结合。

## 4. GraphGAN Framework

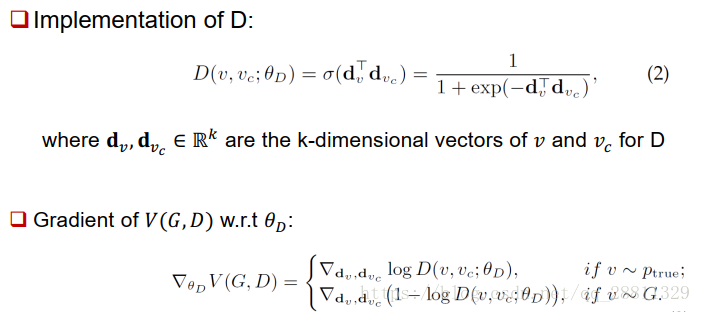
Generator G(v|vc)  tries to fit the underlying true connectivity distribution ptrue(v|vc)，generates the most likely vertices to be connected with vc; Discriminator D(v; vc)  tries to distinguish well-connected vertex pairs from ill-connected ones, outputs a single scalar representing the probability of an edge existing between v and vc.



对于上式，第一项的点是和vc真实相连的点sample出来的，第二项是从G生成的sample出来。给定IMG_256，想minimize这个式子，学习G的参数，使G生成的点尽量像真实分布；给定IMG_257，maximize这个式子，学习D的参数，使得D给真实连接的pair值大，G生成的值小。

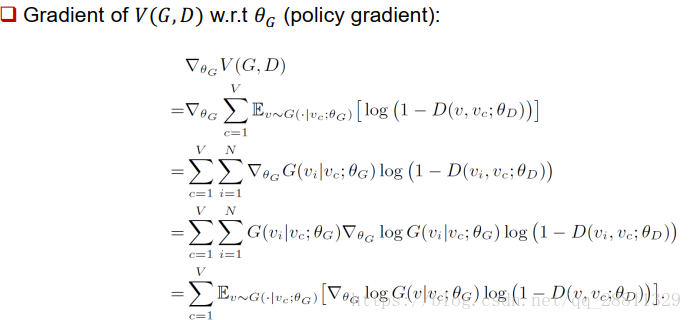
## 5. Discriminator

Given positive samples from true connectivity distribution and negative samples from the generator, the objective for the discriminator is to maximize the log-probability of assigning the correct labels, which could be solved by stochastic gradient ascent. D 定义为输入的两个顶点的内积的sigmoid函数， update only dv and dvc by ascending the gradient.



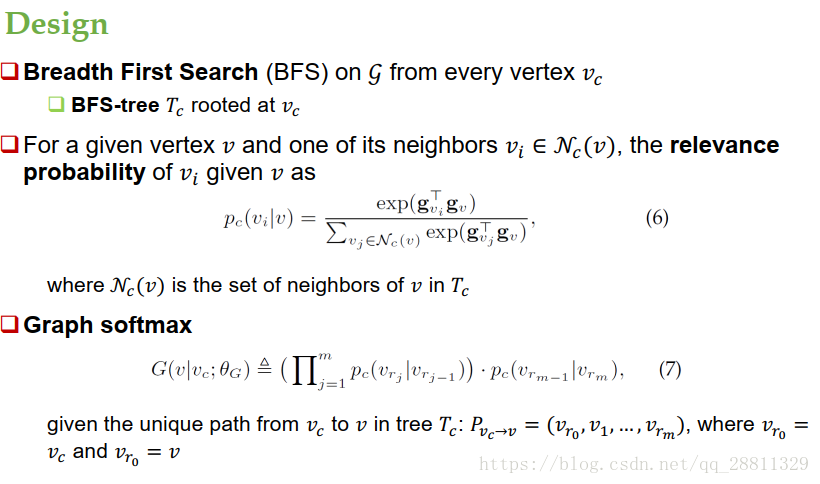
## 6. Generator

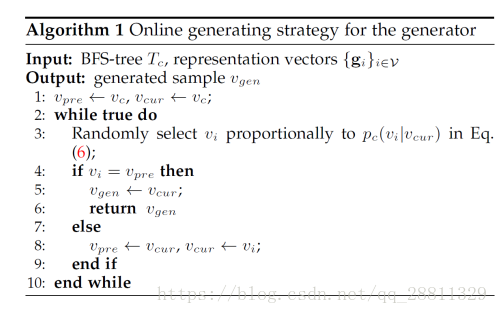
Because the sampling of v is discrete,  we propose computing the gradient of V (G; D) with respect to θG by policy gradient:

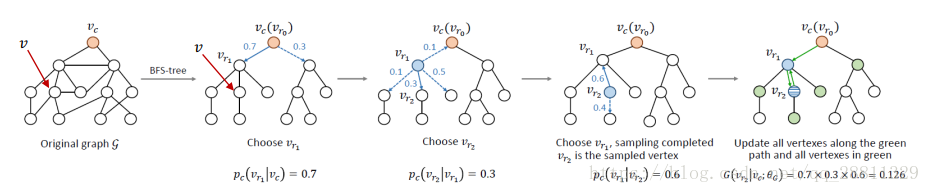


一种最直观的想法是用softmax来实现G，也就是将G(v|VC)定义成一个softmax函数。这种定义有如下两个问题：首先是计算复杂度过高，计算会涉及到图中所有的节点，而且求导也需要更新图中所有节点。这样一来，大规模图将难以适用。 另一个问题是没有考虑图的结构特征，即这些点和Vc的距离未被纳入考虑范围内。

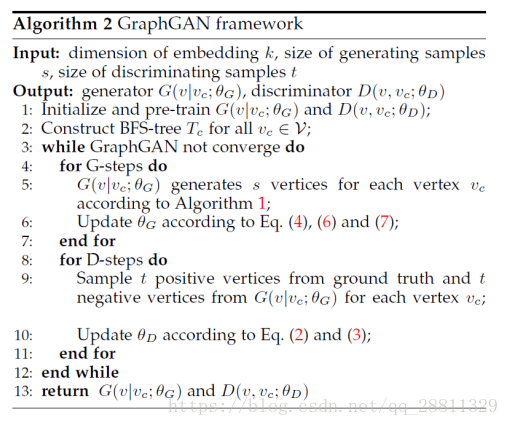
在GraphGAN 中，目标是设计出一种softmax方法，让其满足如下三个要求。第一个要求是正则化，即概率和为 1，它必须是一个合法的概率分布。第二个要求是能感知图结构，并且能充分利用图的结构特征信息。最后一个要求是计算效率高，也就是G概率只能涉及到图中的少部分节点。





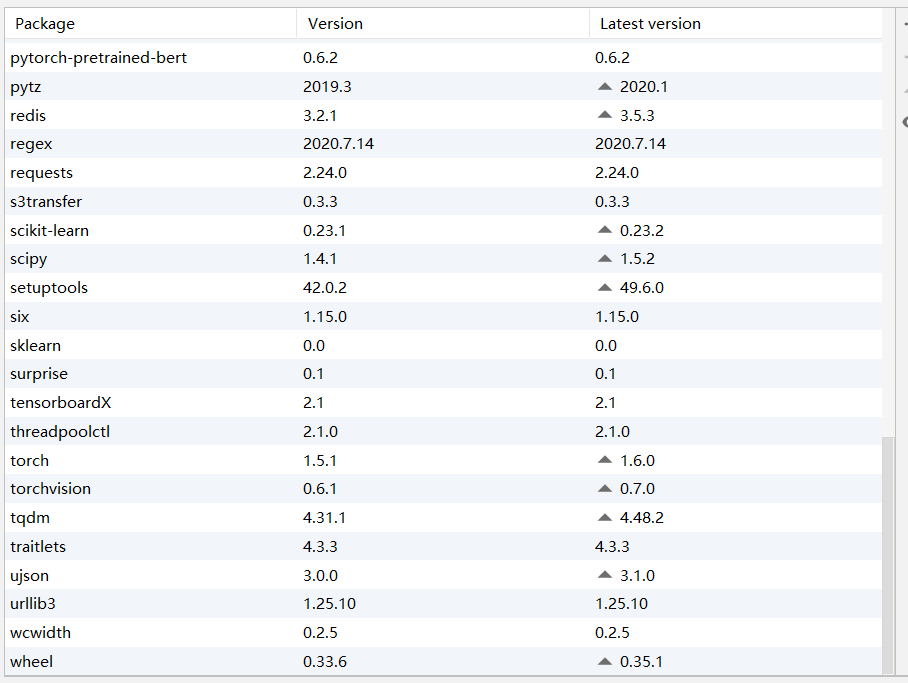


## 7. Algorithm



环境搭建使用pipenv

在pycharm中查看：



训练过程如下：

