Graph Embedding技术之Deepwalk

原创 雷小军 机器学习与数据挖掘 2019-08-26

DeepWalk: Online Learning of Social Representations

Bryan Perozzi
Stony Brook University
Department of Computer Science

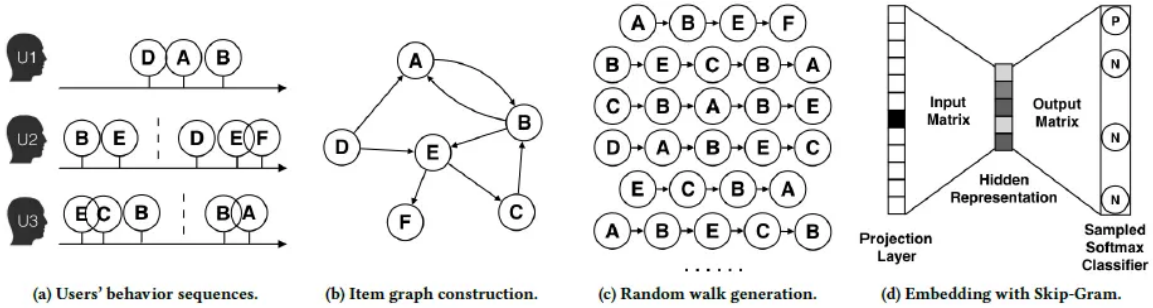
Rami Al-Rfou
Stony Brook University
Department of Computer Science

Steven Skiena
Stony Brook University
Department of Computer Science

{bperozzi, ralrfou, skiena}@cs.stonybrook.edu

前言

自从**word representation**中的神奇算法**word2vec**出现之后，无论是学术界还是工业界都有这样一个共识——**万物皆可Embedding**，基于句子、文档表达的**word2vec、doc2vec**算法，基于物品序列的**item2vec**算法，基于图结构的**Graph Embedding**技术，无论是在推荐、广告还是反欺诈领域，各互联网公司基于自身业务与Embedding结合的论文相继问世，本篇讲述的**Deepwalk**就是**Graph Embedding**技术中的代表，下图是graph embeddings技术的一般流程。

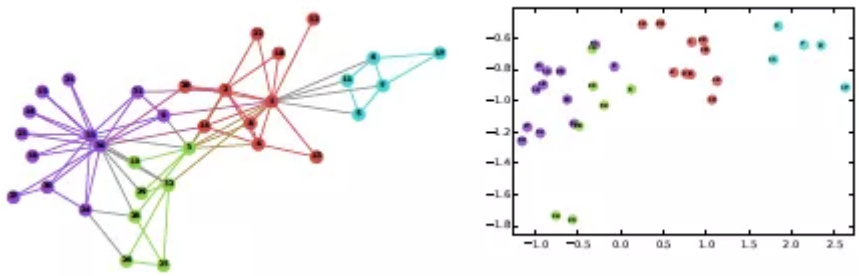


(a) Users' behavior sequences. (b) Item graph construction. (c) Random walk generation. (d) Embedding with Skip-Gram.

DeepWalk论文发表于2014年的KDD会议上，将embedding从item序列推广至图序列，它是一种用于学习网络中顶点的潜在表示的新方法。这些潜在表示将社会关系编码到连续的向量空间中，编码到向量空间后的社会关系，很容易应用到统计模型中。DeepWalk将**随机游走**得到的节点序列当做句子，从截断的随机游走序列中得到网络的局部信息，再通过局部信息来学习节点的潜在表示。

算法思路

DeepWalk将一个图作为输入，并产生一个潜在表示（将图中的每个节点表示为一个向量）作为输出，如下图示例所示：



对于图结构来说，算法设计需要满足以下几个要求：

1、**适应性**：社交网络是不断变化的，当网络发生变化不能对整个网络重新进行计算。

2、**社区意识**：节点的潜在表示对应着维度空间中的距离，应该表示网络中对应的成员的相似度，以此保证网络的同质性。

3、**低维**：当被标记的成员很少时，低维的模型一般表现的更好，并且收敛和推理速度更快。

4、**连续性**：需要通过图的潜在表示来对连续空间中的部分社区成员进行建模。除了提供对社区成员资格的细微视图之外，连续表示还可以使社区之间的决策界限平滑，从而实现更强大的分类。

Deepwalk算法流程

1、展示用户行为序列

2、基于这些用户行为序列构建了物品相关图，图中的边是由用户行为产生的，比如为用户M先后购买了物品A和物品B，会产生了一条有向边由A指向B。其他的有向边也是同样的道理，如果有多个有A指向B的有向边，那么该条边的权重被加强。通过这样的方法将所有用户行为序列都转换成物品相关图中的边后，就得到全局的物品相关图。

3、**采用随机游走的方式随机选择起始点，产生局部物品序列。**

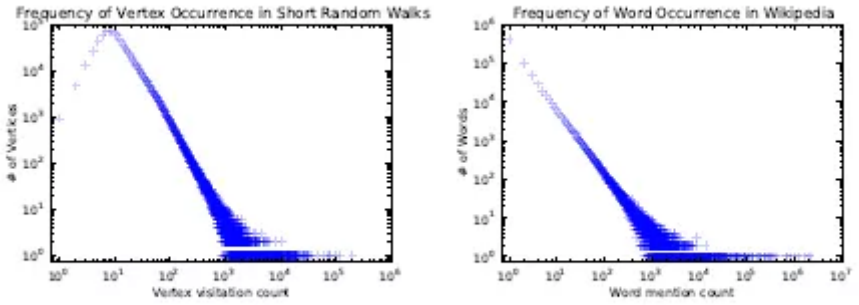
4、将这些物品序列当初句子进行word2vec建模，生成最终的物品Embedding向量。

图中的节点表示item,边表示item之间的交互，上边步骤中最重要的是第三步，如何随机游走产生局部物品序列。deepwalk中的游走是**完全随机**的，这一点需要注意。通过改变**Random Walk**策略才有了后面的**node2vec。**

Random Walk

Random Walk从截断的随机游走序列中得到网络的局部信息，并以此来学习结点的向量表示。借助语言建模word2vec中的一个模型，skip-gram来学习结点的向量表示。将网络中的结点模拟为语言模型中的单词，而结点的序列（由随机游走得到）模拟为语言中的句子，作为skip-gram的输入。

当**图中结点的度**遵循**幂律分布(度数大的节点比较少，度数小的节点比较多**）时，短随机游走中**顶点出现的频率**也将遵循**幂律分布(**即出现频率低的结点多)，又因为**自然语言中单词出现的频率遵循类似的分布**，因此以上**假设可行。**如下图作者针对YouTube的社交网络与Wikipedia的文章进行了研究，得出二者幂率分布基本上类似。



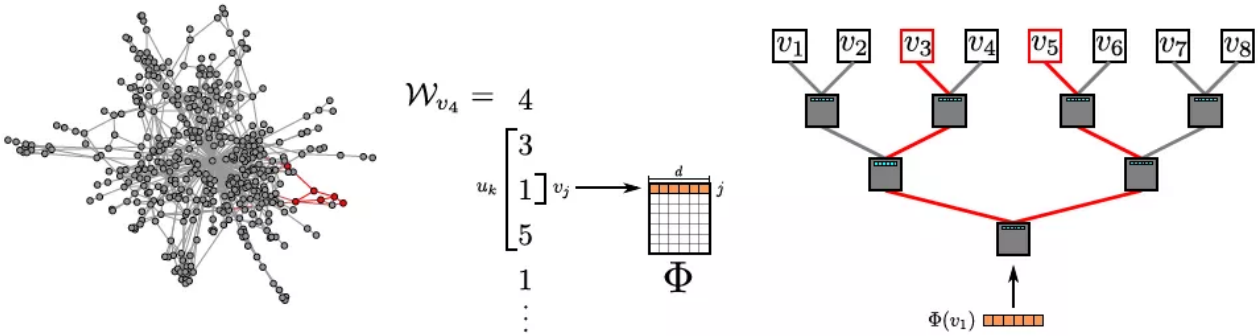
(a) YouTube Social Graph (b) Wikipedia Article Text

优点：

1、并行性：同时进行多个随机游走

2、适应性：当图变化后，不需要全局重新计算，可以迭代地更新学习模型，适合online learning。

算法实现



(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

Deepwalk算法架构

该算法由两部分组成：一个随机游走生成器和一个更新程序。

Algorithm 1 DEEPWALK(G, w, d, γ, t)

Input: graph $G(V, E)$
window size w
embedding size d
walks per vertex γ
walk length t
Output: matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

```
Initialization: Sample Φ from U^{|V|×d}
Build a binary Tree T from V
for i = 0 to γ do
    O = Shuffle(V)
    for each v_i ∈ O do
        W_{v_i} = RandomWalk(G, v_i, t)
        SkipGram(Φ, W_{v_i}, w)
    end for
end for
```

Skip-Gram

SkipGram是一个语言模型，用于最大化句子中出现在窗口w内的单词之间的共现概率。它使用独立性假设，最后条件概率近似为：

$$\Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi(v_i)\right) = \prod_{\substack{j=i-w \\ j \neq i}}^{i+w} \Pr(v_j | \Phi(v_i))$$

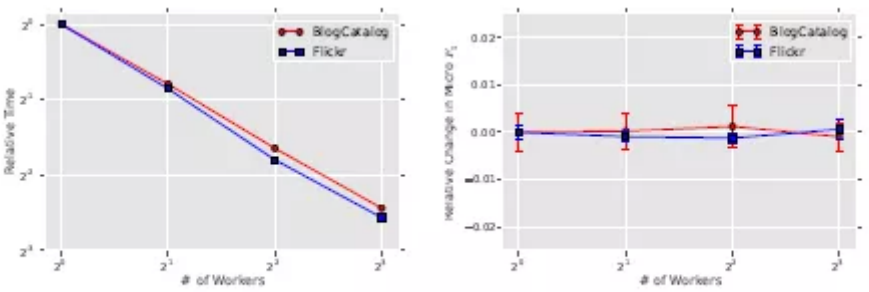
对序列中的每个顶点，**计算条件概率**，即该结点出现的情况下序列中其他结点出现的概率的log值并借助**随机梯度下降算法**更新该结点的向量表示。

Algorithm 2 SkipGram(Φ, $\mathcal{W}_{v_i}$, w)

```
for each v_j ∈ W_{v_i} do
    for each u_k ∈ W_{v_i}[j − w : j + w] do
        J(Φ) = − log Pr(u_k | Φ(v_j))
        Φ = Φ − α ∗ ∂J/∂Φ
    end for
end for
```

注：Φ(vj)为当前结点的向量表示。

并行性



(a) Running Time (b) Performance

上图显示了并行对DeepWalk的影响。图(a)当多个任务同时进行时，算法的速度变快。图(b)表明，并行运行下，DeepWalk的性能没有受到影响。

实验效果展示

数据集

- **BlogCatalog**是博客作者的社交关系网络。标签代表作者提供的主题类别。
- **Flickr**是照片分享网站用户之间的联系网络。标签代表用户的兴趣组，如"黑白照片"。
- **YouTube**是流行的视频分享网站用户之间的社交网络。这里的标签代表喜欢不同类型视频（例如动漫和摔跤）的观众群体。

Name	BLOGCATALOG	FLICKR	YOUTUBE
$\|V\|$	10,312	80,513	1,138,499
$\|E\|$	333,983	5,899,882	2,990,443
$\|\mathcal{Y}\|$	39	195	47
Labels	Interests	Groups	Groups

对比算法

- SpectralClustering
- Modularity
- EdgeCluster
- wvRN
- Majority

实验中通过**多标签分类任务**来评估算法的性能。从数据集中随机抽样标记节点的一部分(TR)，并将其用作训练数据。其余的节点被用作测试。我们重复这个过程10次，并报告Macro-F1和Micro-F1的平均性能。对于所有的模型，使用由LibLinear实现的one-vs-rest逻辑回归扩展来返回最可能的标签。将DeepWalk中的参数设置为：γ=80，w=10，d =128。在SpectralClustering，Modularity，EdgeCluster中，将维度设置为500。

实验结果及分析

(1) BlogCatalog

	% Labeled Nodes	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1(%)	DeepWalk	**36.00**	**38.20**	**39.60**	**40.30**	**41.00**	**41.30**	41.50	41.50	42.00
	SpectralClustering	31.06	34.95	37.27	38.93	39.97	40.99	**41.66**	**42.42**	**42.62**
	EdgeCluster	27.94	30.76	31.85	32.99	34.12	35.00	34.63	35.99	36.29
	Modularity	27.35	30.74	31.77	32.97	34.09	36.13	36.08	37.23	38.18
	wvRN	19.51	24.34	25.62	28.82	30.37	31.81	32.19	33.33	34.28
	Majority	16.51	16.66	16.61	16.70	16.91	16.99	16.92	16.49	17.26
Macro-F1(%)	DeepWalk	**21.30**	**23.80**	25.30	26.30	27.30	27.60	27.90	28.20	28.90
	SpectralClustering	19.14	23.57	**25.97**	**27.46**	**28.31**	**29.46**	**30.13**	**31.38**	**31.78**
	EdgeCluster	16.16	19.16	20.48	22.00	23.00	23.64	23.82	24.61	24.92
	Modularity	17.36	20.00	20.80	21.85	22.65	23.41	23.89	24.20	24.97
	wvRN	6.25	10.13	11.64	14.24	15.86	17.18	17.98	18.86	19.57
	Majority	2.52	2.55	2.52	2.58	2.58	2.63	2.61	2.48	2.62

Table 2: Multi-label classification results in BlogCatalog

在实验中，将BlogCatalog网络上的训练比率(TR)从10%提高到90%，粗体数字表示每列中最高的性能。

结果分析：

1、DeepWalk的性能始终优于EdgeCluster，Modularity和wvRN。DeepWalk在只有20%的节点被标记时的性能，比这些方法在90%的数据时被标记的情况下执行得更好。

2、SpectralClustering的性能更具竞争力，但是当Macro-F1(TR≤20%)和Micro-F1(TR≤60%)上的标记数据稀疏时，DeepWalk仍然表现优异。

通过以上两点可以看出，算法的优势在于，只有小部分图表被标记时，具有强大的性能。

(2) Flickr

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**32.4**	**34.6**	**35.9**	**36.7**	**37.2**	**37.7**	**38.1**	**38.3**	**38.5**	**38.7**
	SpectralClustering	27.43	30.11	31.63	32.69	33.31	33.95	34.46	34.81	35.14	35.41
	EdgeCluster	25.75	28.53	29.14	30.31	30.85	31.53	31.75	31.76	32.19	32.84
	Modularity	22.75	25.29	27.3	27.6	28.05	29.33	29.43	28.89	29.17	29.2
	wvRN	17.7	14.43	15.72	20.97	19.83	19.42	19.22	21.25	22.51	22.73
	Majority	16.34	16.31	16.34	16.46	16.65	16.44	16.38	16.62	16.67	16.71
Macro-F1(%)	DeepWalk	**14.0**	17.3	**19.6**	**21.1**	**22.1**	**22.9**	**23.6**	**24.1**	**24.6**	**25.0**
	SpectralClustering	13.84	**17.49**	19.44	20.75	21.60	22.36	23.01	23.36	23.82	24.05
	EdgeCluster	10.52	14.10	15.91	16.72	18.01	18.54	19.54	20.18	20.78	20.85
	Modularity	10.21	13.37	15.24	15.11	16.14	16.64	17.02	17.1	17.14	17.12
	wvRN	1.53	2.46	2.91	3.47	4.95	5.56	5.82	6.59	8.00	7.26
	Majority	0.45	0.44	0.45	0.46	0.47	0.44	0.45	0.47	0.47	0.47

Table 3: Multi-label classification results in Flickr

在实验中，将Flickr网络上的训练比率(TR)从1%变为10%。这相当于在整个网络中有大约800到8000个节点标记用于分类。上表给出了实验结果，粗体数字表示每列中最高的性能。

结果分析：

1、对于Micro-F1，DeepWalk的性能至少要比其他算法高出3%。DeepWalk可以比其他算法少60%的训练数据。

2、在Macro-F1中表现也相当不错，SpectralClustering最接近它。

（3）YouTube

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**37.95**	**39.28**	**40.08**	**40.78**	**41.32**	**41.72**	**42.12**	**42.48**	**42.78**	**43.05**
	SpectralClustering	—	—	—	—	—	—	—	—	—	—
	EdgeCluster	23.90	31.68	35.53	36.76	37.81	38.63	38.94	39.46	39.92	40.07
	Modularity	—	—	—	—	—	—	—	—	—	—
	wvRN	26.79	29.18	33.1	32.88	35.76	37.38	38.21	37.75	38.68	39.42
	Majority	24.90	24.84	25.25	25.23	25.22	25.33	25.31	25.34	25.38	25.38
Macro-F1(%)	DeepWalk	**29.22**	**31.83**	**33.06**	**33.90**	**34.35**	**34.66**	**34.96**	**35.22**	**35.42**	**35.67**
	SpectralClustering	—	—	—	—	—	—	—	—	—	—
	EdgeCluster	19.48	25.01	28.15	29.17	29.82	30.65	30.75	31.23	31.45	31.54
	Modularity	—	—	—	—	—	—	—	—	—	—
	wvRN	13.15	15.78	19.66	20.9	23.31	25.43	27.08	26.48	28.33	28.89
	Majority	6.12	5.86	6.21	6.1	6.07	6.19	6.17	6.16	6.18	6.19

Table 4: Multi-label classification results in YouTube

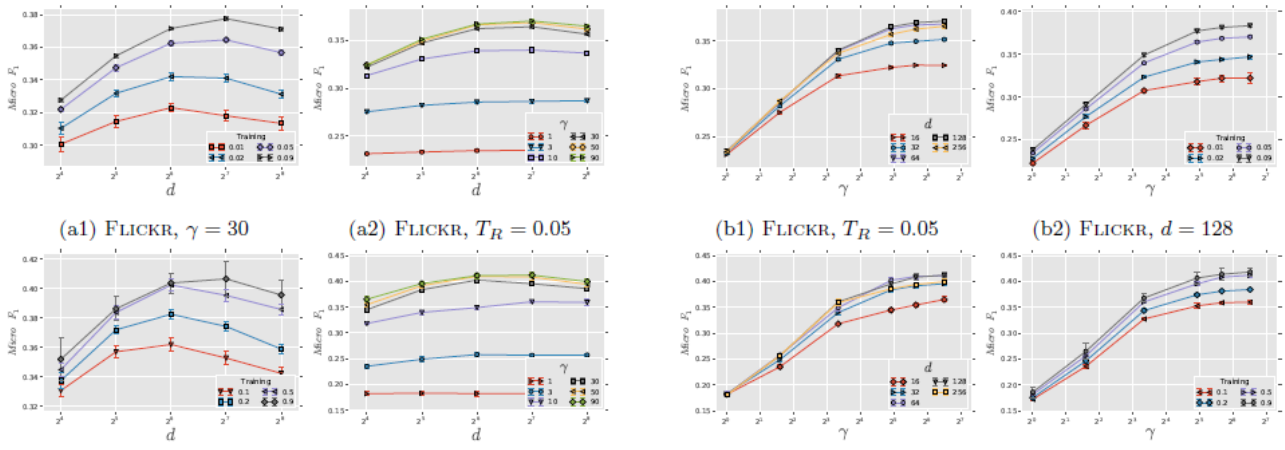
YouTube网络规模大，更接近现实世界网络。SpectralClustering和Modularity不能用于这种规模的网络。在实验中，训练比率(TR)从1%变化到10%，粗体数字表示每列中最高的性能。

结果分析：

从实验中可以看出，DeepWalk明显优于其他算法：DeepWalk可以扩展到大图，并且在这样一个稀疏标记的环境中执行得非常好。

参数敏感度实验

为了评估DeepWalk的参数变化如何影响其在分类任务上的性能，我们对两个多标签分类任务(Flickr和BlogCatalog)进行了实验。实验中固定窗口大小和步长(w = 10，t = 40)，然后，改变嵌入维度(d)，每个游走长度(γ)，以及可用的训练数据量(TR)，以确定它们对网络分类性能的影响。



(a1) FLICKR, $\gamma = 30$ (a2) FLICKR, $T_R = 0.05$ (b1) FLICKR, $T_R = 0.05$ (b2) FLICKR, $d = 128$

(a3) BLOGCATALOG, $\gamma = 30$ (a4) BLOGCATALOG, $T_R = 0.5$ (b3) BLOGCATALOG, $T_R = 0.5$ (b4) BLOGCATALOG, $d = 128$

(a) Stability over dimensions, d (a) Stability over number of walks, γ

Figure 5: Parameter Sensitivity Study

图(a)显示了维度变化的影响。图(a1)和(a3)分析了维度和训练数据比例对性能的影响。图(a2)和(a4)展示了维数和随机游走长度对性能的影响。得到如下结论：

1、a1和a3显示模型的最佳维度取决于训练数据的数量。

2、通过a2和a4可以看出，在γ确定的情况下，不同维度下，算法的性能是相对稳定的。当γ>=30时，算法的性能比较好。两个图在γ的不同值之间的相对差异是一致的。

这些实验表明，算法可以生成各种大小的有用模型。模型的性能取决于**随机游走的数量**，而模型的**最优维度**取决于可用的训练样例。

图(b)显示了改变γ对性能的影响。结果对于不同的维度(图(b1)，图(b3))和不同训练数据量(图(b2)，图(b4))非常一致。增加γ对结果有很大的影响，但是当γ>10时，这种影响迅速减慢。这些结果表明，当**随机游走的数量足够多**时，我们才能够学习结点的有意义的潜在表示。

读了这篇文章给大家留几个问题：

（1）你觉得Deepwalk中的随机游走方式合理吗？

（2）来了一个新的item，怎么求它的embedding，需要重新训练吗？

总结

最近在重温embedding相关的论文，每一次看论文都有不同的感受，graph embedding已经广泛应用在推荐、广告、风控反欺诈等领域，值得我们好好读一读，有什么问题可以和我交流。