DeepWalk: Online learning of Social Representations

原创 村头陶员外 跟我一起读论文啦啦 2019-10-30

本次要分享的是 14 年论文 DeepWalk: Online learning of Social Representations， 论文链接**DeepWalk**[1]，参考的代码**CODE**[2]，本论文是图表示学习领域内的一篇较早的文章，是学习图表示学习绕不过的一篇文章，虽然整体难度不大，但是文章所提出的方法个人感觉非常独到和有趣。

论文动机和创新点

- 本论文提出的是一种针对图结构的**representation learning 方法**，其从某种角度来说机器学习或深度学习的发展就是围绕着**representation learning 方法**进行的。

- 在自然语言处理领域，word2vec（**以前总结的 word2vec**[3]）是一个非常基础和著名的词表示方法，利用 **句子** 中词与词之间的共现或相邻关系，对词进行表示学习，所学习到的词表示向量能准确的刻画词与词之间的实际意义关系。

- 那么在非语言结构的其他结构中，例如图结构中，是否可以根据图中节点与节点相邻关系，来学习每个节点的表示呢？显然是可以的，论文中提出，在图结构中，以任意一个节点为起始节点，进行随机游走，当游走到最长的步数时，可以获取一串由节点构成的序列，这个序列就可以类比自然语言中的句子，节点类比句子中的词，然后利用 word2vec 的方法对每个节点进行表示学习。

- 该论文所提出的方法是一种无监督的特征学习方法，具体来说，就是从可截断的随机游走中得到一串节点序列，利用 word2vec 方法学习每个节点的表示向量。我们认为这样学习到的表示向量可以捕捉到**节点之间的邻近相似关系以及其所属社区（类别）的关系。**

- 论文中讲到所提方法有以下特点 ① 适应性：在实际的社交网络中，由社交关系产生的图是不断发展的，而该方法可以自适应的去学习，不必重复从头再学习。② 社区意识：该方法学习到的隐空间应该表示网络中同质节点或相邻节点的距离远近信息。③

低维度：当带标签的数据稀疏和低维度时，模型生泛化能力更好，训练更容易收敛。
④ 连续性：该方法所学习到的表示是连续型的，因此具有光滑的决策边界，鲁棒性更高。

- 论文实验证明了，在缺乏足够的带标签数据时，本文所提方法比其他的无监督表征学习方法效果上更好。

- 个人感觉，本论文在实验部分，某些细节并未讲清，例如图是如何构建的？节点和边的具体业务意义是什么？并未讲清，只是笼统的说，利用这些数据构图，有多少个节点，多少条边。

- 在实际应用时，需根据业务逻辑，定义节点，以及根据某种规则定义连接节点的边，由此可以构成一张图进行表示学习。

Learning Social Representations

图的定义

在社交网络中多分类任务为例：

$$G = (V, E)$$

V 表示图中的节点，节点的含义为图中的类别(member)。E 表示图中的边，$E \subseteq (V \times V)$ $G_L = (V, E, X, Y)$ 表示一个带标签的图，其中$X \in R^{|V| \times S}$， S 表示节点的特征空间大小，$Y \in R^{|V| \times \gamma}$， γ 表示类别个数。

在实际应用中，是需要根据业务的具体逻辑去构建图。传统的机器学习方法是由 X 映射到 Y，模型所需要学习的是如何准确的学习到这个映射函数。而**本论文所提方法是独立于标签的分布信息，由图中的拓扑信息去学习节点的向量表示**。是完全无监督学习方法。

该方法所学习到的表示向量可以使用在任何的分类算法中。

随机游走

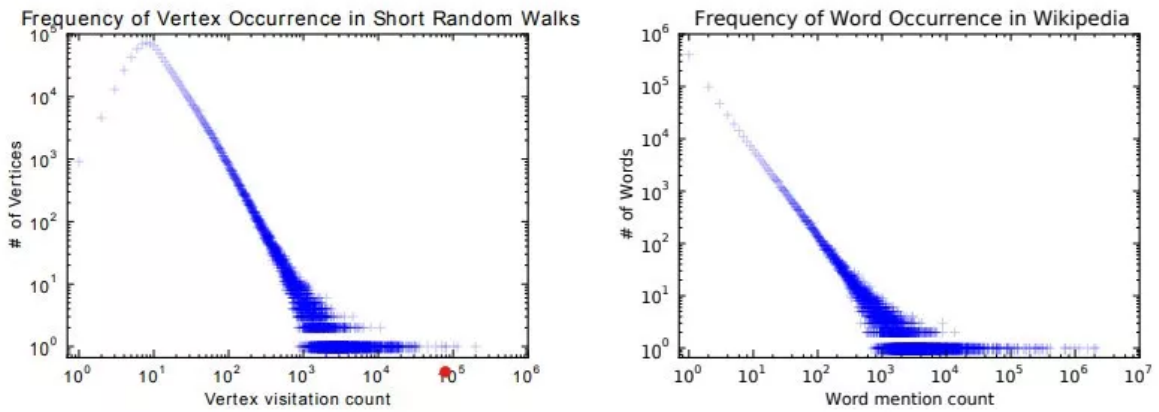
以任意一个节点为起始节点，每次随机的选择与它邻近节点进行游走，直到达到最大步长。这种截断的随机游走方式提供了两个优点

- 局部探索游走很容易并行化进行，几个不同 walker 可以同时游走多条不同的线路。
- 从截断随机游走中获取信息，当图结构发生小的变化时，不需要重复重新的去学习。

在实际应用时，方便很多。

长尾分布

在论文指的是 power laws，其实和长尾分布大同小异，在自然语言领域，我们发现大部分词的词频都很小，只有少数词的词频很高，符合长尾分布。而在 YouTube Social Graph 中，进行随机游走，发现节点的分布也是符合这种长尾分布的，如下图所示：



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

Figure 2: The distribution of vertices appearing in short random walks (2a) follows a power-law, much like the distribution of words in natural language (2b).

词频分布和节点频率分布一致，因此认为在自然语言处理领域内有效的 word2vec 方法可以复用在图结构中。

Deep Walk

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 $\mathcal{U}^{|V| \times d}$
Build a binary Tree T from V
for $i = 0$ to γ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  for each $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, \text{t})$
    SkipGram(Φ, $\mathcal{W}_{v_i}$, w)
  end for
end for

在这里插入图片描述

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

for each $v_j \in \mathcal{W}_{v_i}$ **do**
  for each $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ **do**
    $J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$
    $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$
  end for
end for

以上两个算法流程已经能很好的说明 Deep Walk 的步骤，其中 SkipGram 算法表示由某个节点预测他周围的节点，是 word2vec 方法中的一种模型。

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

上述公式表示，由节点v_i来预测与之相邻的周围节点v_j，**其中所得到的副产品Φ这个embedding矩阵就是我们要学习的目标。**

和自然语言处理相同，这里的 skip-Gram 同样面临计算复杂度很高的问题，如果不做任何处理，每次都需要对所有节点进行概率计算，时间复杂度过高，训练缓慢，由此引入了 Hierarchical Softmax。

Hierarchical Softmax 是 NLP 中常用方法，详情可以查看**Hierarchical Softmax[4]**。其主要思想是以词频构建 Huffman 树，树的叶子节点为词表中的词，相应的高频词距离根结点更近。当需要计算生成某个词的概率时，不需要对所有词进行 softmax 计算，而是选择在 Huffman 树中从根结点到该词所在结点的路径进行计算，得到生成该词的概率，时间复杂度从 O(N) 降低到 O(logN)（N 个结点，则树的深度 logN）。该方法同样适用在图结构的 skip-Gram 模型中。

实验

论文中使用了三个数据集：

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

但是论文没有详细给出在三个数据集中，所构成的图结构中节点以及边的实际业务意义，例如在图中，每个数据集中节点具体指的是什么？以什么样的规则来定义边的？

论文实验中，随机抽取部分样本作为训练集（带 label），剩余的作为测试集，在训练集中，以上面所说的方式（DeepWalk+Skip-Gram+Hierarchical softmax）学习每个节点的表示向量，然后以学习到的表示向量作为特征，LR 分类器进行训练，训练出一个分类模型；再在测试集上进行测试。实验结果如下：

	% 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

	% 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

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
ficro-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
facro-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

可以看出当选取**较少的带 label 训练数据时**（label sparse），相对于其他方法，DeepWalk 的效果更好。

个人总结

本文所提方法不难，其 word2vec 也是很基础的方法，但是能借此应用到图结构中的确很有趣和有意义，使得 word2vec 这种简单有效的方法能应用的更加广泛。

参考资料

[1] **DeepWalk:** http://www.perozzi.net/publications/14_kdd_deepwalk.pdf

[2] **CODE:** https://github.com/phanein/deepwalk

[3] 以前总结的**word2vec:** https://blog.csdn.net/Mr_tyting/article/details/80091842

[4] **Hierarchical Softmax:** https://blog.csdn.net/itplus/article/details/37969979