Finding Seeds and Relevant Tags Jointly: For Targeted Influence Maximization in Social Networks

背景

人们可以通过口碑交流的方式在社交网络上快速传播信息，营销商可以利用社交网络平台通过一系列手段选取影响力较大的用户作为种子节点，让他们的产品尽可能推销到多的用户面前，从而赚取利益。这种市场营销策略可以被形式化为影响力最大化问题，即选择k个种子节点，使其影响力传播期望最大化。而比较实际的情况是，营销商根据产品属性可以确定他们的目标用户，而不是所有用户，即目标影响力最大化问题。该文基于此点，并考虑到社交网络中的用户通常应用标签属性来描述自身信息，得到了一个最优种子集合和最优标签集合的联合优化问题。

相关工作

- IM问题，Kempe等人形式化提出。通过爬山法得到$1-1/e$近似保证；之后被改进，像CELF和CELF++,通过偷懒估值法来减少计算时间；MIA算法通过最大化路径概率计算影响传播；TIM/TIM+和IMM，基于RIS采样技术，进一步提高效率和有效性。其边传播概率是固定的。
- 基于话题的IM问题，在给定话题集合下，选择种子集合使其影响力传播期望最大化。Barbieri等人确定种子节点同时寻找最优话题分布，而实际情况下很难确定一个最优的话题分布。
- 条件可靠性研究，主要在系统和设备网络中，他们定义源节点到达目标节点的边存在概率为关于内在因素的条件概率。该文设计边传播概率基于这种思想。

解决问题

传统影响力最大化问题，边上的传播概率是固定的，选择k个种子节点在整个图上最大化影响传播。而实际上边概率取决于边信息，营销商也有自己确定的目标用户。该文通过用户标签来计算边概率，并将其设计为一个目标影响力最大化问题，更加适应实际情境。

创新之处

- 提出了一个新的问题，联合优化选择 k 个种子节点和 $top-r$ 个标签，使得在目标用户集合上影响力传播期望最大化。
- 通过智能索引方法创建图减少时间消耗，设计批次路径选择算法确定 $top-r$ 标签，通过迭代算法计算影响传播。

解决方法

问题定义

社交网络由一个不确定图$\mathcal{G}=(V,E,P)$表示，V是节点集合，E是边集，P是边上的概率集合。对于条边$e=(u,v)\in E$，每个标签$c\in C$，$P((u,v)|c)$表示一个激活的用户u影响到用户v的概率。

传播模型采用独立级联模型，并使用等价的概率图形式。即概率图$G=(V,E_G)$为$\mathcal{G}$的确定实例，$E_G\subseteq E$。加入标签后，就变成基于标签意识的IC模型，在给定标签集合$C_1\in C$下，边e的信息传播概率为$P(e|C_1)=\mathbb{F}_{c\in C_1}(P(e|c))$，$\mathbb{F}$为聚合函数，这里采用标签影响概率独立的形式，即

$$P(e|C_1) = 1 - \prod_{c \in C_1} (1 - P(e|c))$$

那么概率图G的概率就是

$$Pr(G|C_1) = \prod_{e \in E_G} P(e|C_1) \prod_{e \notin E_G} (1 - P(e|C_1))$$

那么问题就表述为寻找k各种子节点集合S和$top-r$标签集合C_1，使其影响传播期望$\sigma(S,T,C_1)$最大化：

$$\sigma(S,T,C_1) = \sum_{G \subseteq \mathcal{G}|C_1} [\sigma(S,T) \times Pr(G|C_1)]$$

H3 智能概率图索引

将每个概率子图用I表示，并通过标签进行索引，即一个概率图索引表示为(I,c)。它由以下步骤生成：

1. 只保存与标签 c 有关的边；
2. 移除掉任何满足概率 $1-p(e|c)$ 的边 $e \in E$；
3. 保留所有 G 中的节点，即使存在没有相连边的节点。

建立概率图索引之后，按照选择的$top-r$标签的r进行组合，即将r个概率图索引进行组合，然后通过选择随机目标节点，利用深度优先搜索寻找与之相连的节点集合，作为RR集合。

上述办法是可以进行优化的，在迭代过程中，已经建立的概率图索引就无需再建立，只对未出现的标签进行建立概率图索引。再就是反向BFS寻找RR集合时，不需要一直找到最后一个节点，因为实际上，目标节点是存在有关标签属性的局部聚类的，即某块区域都是关于某个标签内容的目标节点，因此可以设置一个步数h来规定反向BFS寻找的深度。这种建立概率图索引的方法是lazy和local的，因此作者称之为LL-TRS Index，它更节约时间和内存。

H3 批次路径选择算法

通过建立一个批次路径格$\mathcal{P}(C)$，应用贪心算法来寻找最优标签集合。可以将其看成一棵树，树的节点索引为用户节点到达目标节点边上的标签种类，节点内容为对应的路径$P \in \mathcal{P}(C)$。

假设用$\mathcal{PB}$表示所有批次路径的集合，$\mathcal{PB}'$表示已经被选择的路径的集合，$Des\mathcal{P}$表示批次路径$\mathcal{P}$的子代，那么批次路径选择算法描述为：

1. 通过最大化边际增益率 $\mathcal{P}^* = argmax_{\mathcal{P} \in \mathcal{PB} \setminus \mathcal{PB}'} [\frac{\sigma(S,T,Des\mathcal{P} \cup \mathcal{PB}') - sigma(S,T,\mathcal{PB}')}{C(\mathcal{P})}]$ 来选择最优路径批次 $\mathcal{P}^*$ 和它的后代加入 $\mathcal{P}_1$
2. 将 $\mathcal{P}_1$ 中包含的标签加入到 C_1
3. 移除掉原批次路径格中所有关于该标签的批次路径
4. 直到选够 r 个标签为止，返回 C_1

H3 联合查找K-seeds和r-Tags

- 贪心算法

  首先在所有标签集下寻找最优种子节点，然后通过这个种子节点和所有目标节点，运用批次路径选择算法纠正标签集合。然后继续这样做下去，直到找到k个种子节点和r个标签集合。

- 迭代收敛法

  1. 通过一定方法初始化种子节点集合和标签集合
  2. 在给定标签集合情况下贪心寻找最优种子节点集合，在该集合下通过批次路径选择算法调整标签集合
  3. 直到算法收敛为止，返回找到的 S^* 和 C^*

  初始化方法有三种：

  - 都采用均匀随机选择

- 考虑所有标签集合，通过贪心算法寻找 k 个最优种子节点组成的集合 S
- 聚合出现在所有目标节点入边的标签，选择 $top-r$ 个累积概率最高的标签作为初始标签集。

H2 实验

H3 数据集

Name	#Nodes	#Edges	#Tags	Edge Prob: Mean, SD, Quartiles
lastFM	1 322	13 842	78	0.26, 0.23, {0.06, 0.2, 0.41}
DBLP	704 266	4 727 290	230	0.26, 0.15, {0.18, 0.18, 0.33}
Yelp	125 368	808 909	195	0.33, 0.25, {0.18, 0.26, 0.5}
Twitter	6 294 565	11 063 034	500	0.27, 0.14,{0.18, 0.18, 0.33}

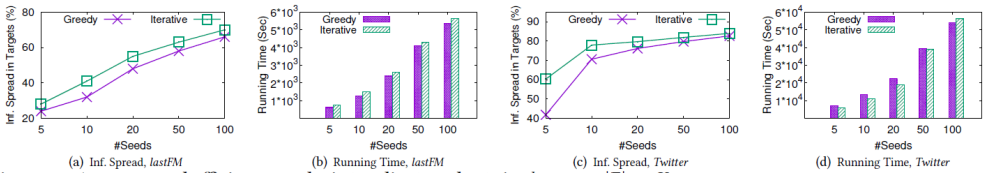
- LastFM：音乐收听历史数据集
- DBLP：合作网络数据集
- Yelp：用户业务评论数据集
- Twitter：社交平台数据集

H3 参数设置

种子集合大小$5-100$，标签集合大小$5-30$，目标集合大小$500-LastFM$，$3K-Other$，局部步长$h=3$。

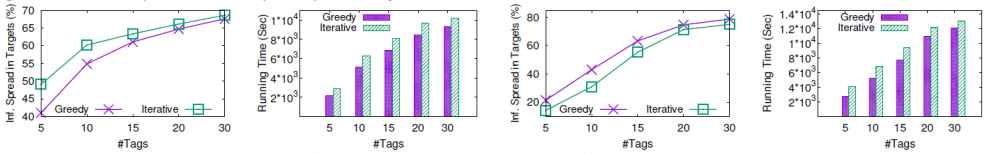
算法：贪心算法和迭代算法，对于迭代算法考虑4中初始化方式：随机初始化种子和标签（RS，RT），基于IM的种子选择（IMS），基于标签频率初始化（FT）。

H3 结果



(a) Inf. Spread, *lastFM* (b) Running Time, *lastFM* (c) Inf. Spread, *Twitter* (d) Running Time, *Twitter*

Figure 13: Accuracy and efficiency analysis to adjust seed set size k. $r=10$, $|T|=3K$.



(a) Inf. Spread, *DBLP* (b) Running Time, *DBLP* (c) Inf. Spread, *Yelp* (d) Running Time, *Yelp*

Figure 14: Accuracy and efficiency analysis to adjust tag set size r. $k=20$, $|T|=3K$.

上图分别是贪心算法和迭代算法在四种数据集上影响传播和运行时间的表现。总体上看，迭代算法取得的影响传播要高于贪心算法，而二者运行时间相差不大。

Method	Inf. Spread in Targets (%) # Seeds (k) 5	10	20	50	100	Running Time (Sec) # Seeds (k) 5	10	20	50	100
RS+RT	42	64	75	82	84	6 400	11 100	17 100	27 200	39 400
IMS+RT	42	66	75	82	84	5 800	9 500	14 800	23 500	34 200
RS+FT	48	74	78	82	84	**3 700**	**7 700**	**12 000**	**20 100**	**30 200**
IMS+FT	**49**	**74**	**78**	**82**	**84**	5 000	9 200	14 000	22 300	33 100

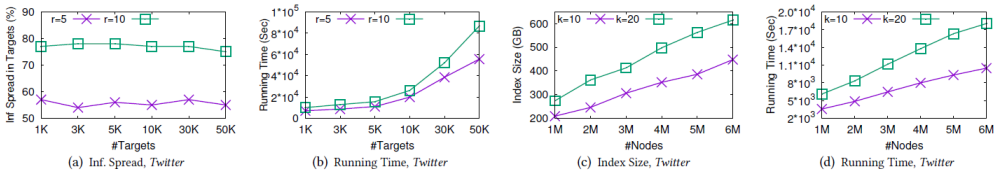
Table 5: Accuracy and efficiency for different initialization methods. *Yelp*. $r=20$, $|T|=3K$.

上表是迭代算法不同初始化方法以及种子集合大小对影响传播和运行时间的影响，可以看出RS+FT初始化方法最优。

Method	Inf. Spread in Targets (%)							
	# Iterations							
	1	1.5	2	2.5	3	3.5	4	4.5
RS+RT	17	42	53	60	63	**64**	**64**	**64**
IMS+RT	42	56	62	**66**	**66**	**66**	converged	converged
RS+FT	60	68	**74**	**74**	**74**	converged	converged	converged
IMS+FT	**64**	**71**	**74**	**74**	**74**	converged	converged	converged

Table 6: Influence spread achieved after various iterations, with different initialization methods, *Yelp*. $k = 10$, $r = 20$, $|T| = 3K$.

这是四种初始化方法在固定种子集合大小情况下，影响传播随迭代次数的变化，可以看出RS+FT和IMS+FT在3次就已经收敛。



(a) Inf. Spread, *Twitter* (b) Running Time, *Twitter* (c) Index Size, *Twitter* (d) Running Time, *Twitter*

Figure 15: Scalability analysis. $k = 10$, $r = 10$.

上图是观察了迭代算法的扩展性，在最大的Twitter数据集上实验，考虑r和k的变化对影响传播和运行时间的影响以及随着节点数增加，索引占内存大小和运行时间的变化。可以看出，时间变化都基本为线性，所以该算法具有很好的扩展性。