*TextRank*算法

The TextRank Algorithm

目录

引言

Introduction

当今，我们处在一个信息爆炸的时代，每天无论是新闻*App*还是微信公众号都在给我们推送着大量我们感兴趣的内容。但是，每个人的时间都是有限的，无法对全部文章都进行浏览。所以，使用一个高效的算法来提取文章的关键词亦或者根据文章内容自动生成摘要，对于这个快节奏的社会来说，是必要的。它能快速帮助我们把握文章内容，过滤掉不重要的信息，将我们的精力花在“刀刃上”。

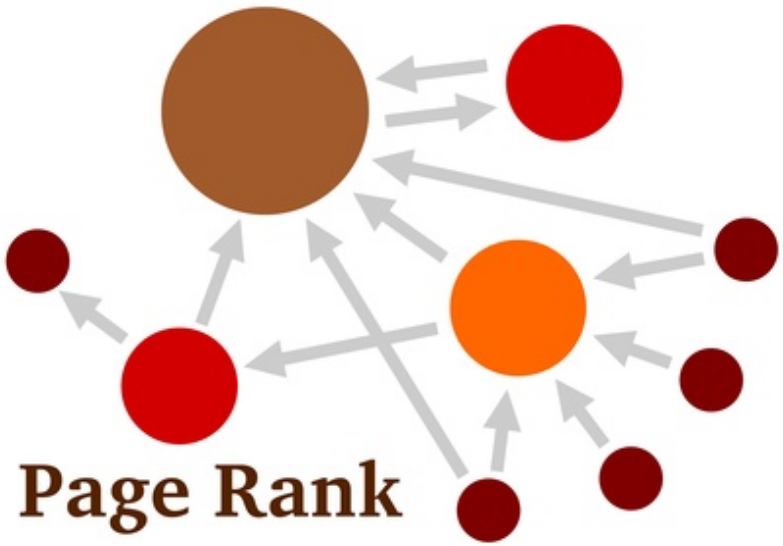
目录

从*PageRank*算法开始谈起

PageRank Algorithm Review

*PageRank*是*Larry Page*在1998年提出的用于*Google*网页重要性排序的算法。

这个算法的基本思想是：将互联网中的网站看作是图的节点，网站间的链出/链入作为节点间连接的边。最终，整个互联网被抽象成一个有向不带权图，通过公式迭代，最终计算出每个网站的*PR*值，*PR*值高的排在前，反之，排在后。核心思想类似于民主投票思想。



从*PageRank*算法开始谈起

PageRank Algorithm Review

- 首先，根据网页链出/链入关系将互联网抽象成有向无权图G

接着，根据图G，计算出每一个节点的概率转移矩阵M，并初始化PR值矩阵R

- 然后，根据以下公式迭代，更新每一个节点的PR值

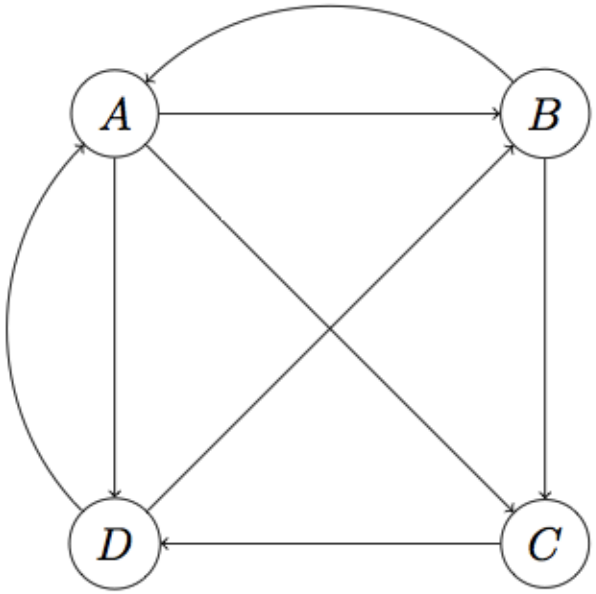
$$PR(i) = d \sum_{j \in in(i)} \frac{PR(j)}{|out(j)|} + (1-d)\frac{1}{N}$$

其中，d为阻尼系数，一般取0.85；N为网站数量，$PR(j)$表示第j个节点的PR值，$|out(j)|$表示第j个节点的链出数

- 最后，当每个网站的PR值基本不在变化时，表示已收敛，算法结束

从*PageRank*算法开始谈起

PageRank Algorithm Review



$$M = \begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 0 & 1 & 0 \end{bmatrix}$$

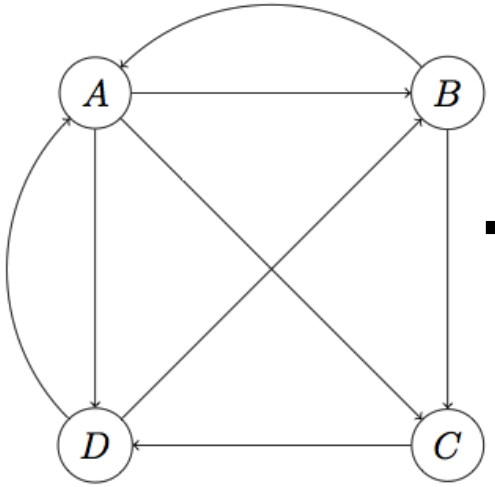
概率转移矩阵

$$\vec{Rank} = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$

*PR*矩阵

从*PageRank*算法开始谈起

PageRank Algorithm Review



$$M * R = R$$

$$\begin{bmatrix} 0 & 1/2 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 1/4 \\ 5/24 \\ 5/24 \\ 1/3 \end{bmatrix}$$

$$[R, N] * \begin{bmatrix} d \\ 1-d \end{bmatrix} = R$$

$$\begin{bmatrix} \frac{1}{4} & 1/4 \\ \frac{5}{24} & 1/4 \\ \frac{5}{24} & 1/4 \\ \frac{1}{3} & 1/4 \end{bmatrix} * \begin{bmatrix} 0.85 \\ 1-0.85 \end{bmatrix} = \begin{bmatrix} 0.25 \\ 0.21 \\ 0.21 \\ 0.32 \end{bmatrix}$$

目录

关键字提取——共现关系

Keyword Extraction——Co-Occurrence

“共现”指文献的特征项描述的信息共同出现的现象，这里的特征项包括文献的外部和内部特征，如题名、作者、关键词、机构等。比如，现有三个段落的分词结果如下：*a/b/c*，*a/b/f*，*a/d/c*，那么就是*ab*共现2次，*ac*共现2次，以此类推。

在文献计量学中，关键词的共现方法常用来确定该文献集所代表学科中各主题之间的关系。例如，需要通过分析一篇小说或剧本，来分析剧中各个角色之间的人物关系，可以用共现关系。

关键字提取——滑动窗口

Keyword Extraction——Slide Window

滑动窗口概念的引入是*TextRank*算法的核心思想之一。通过滑动窗口大小的控制，我们可以操纵词之间的共现关系，进而控制最后生成图的稠密程度，最终决定关键词提取的准确性。

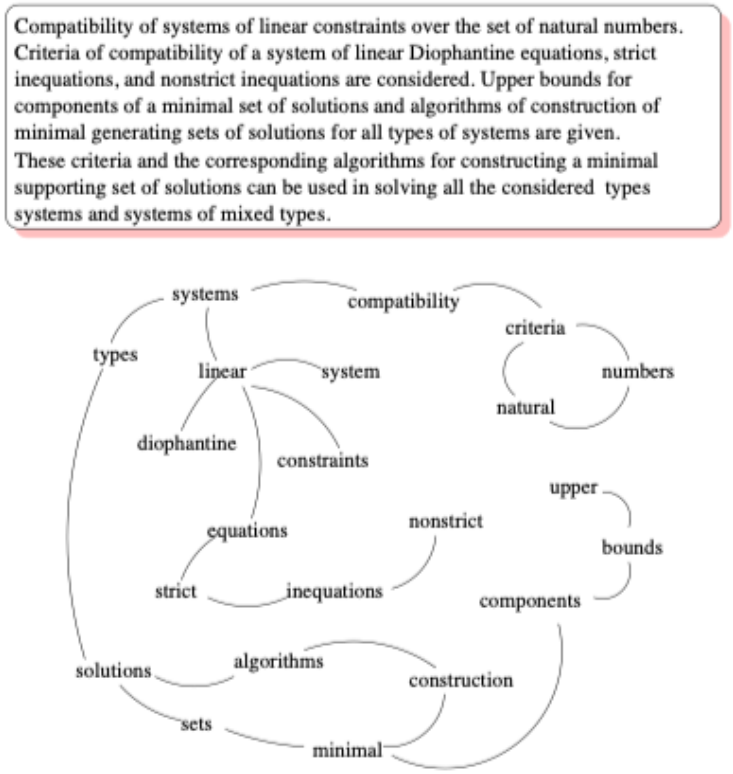
滑动窗口大小越小，最后结果越精确。

关键字提取——图构建

Keyword Extraction——Graph Construction

在*TextRank*中，对于关键词提取应用，我们最后需要构建一个无向无权图。在构建图的过程中，我们需要利用滑动窗口机制和词之间的共现关系。

这里的共现关系，是指共同出现，也就是说，对于图中任意两个节点，它们之间存在边，当且仅当它们所对应的关键词在滑动窗口中共同出现。



图截至《*TextRank:Bringing Order into Texts*》

关键字提取——图构建

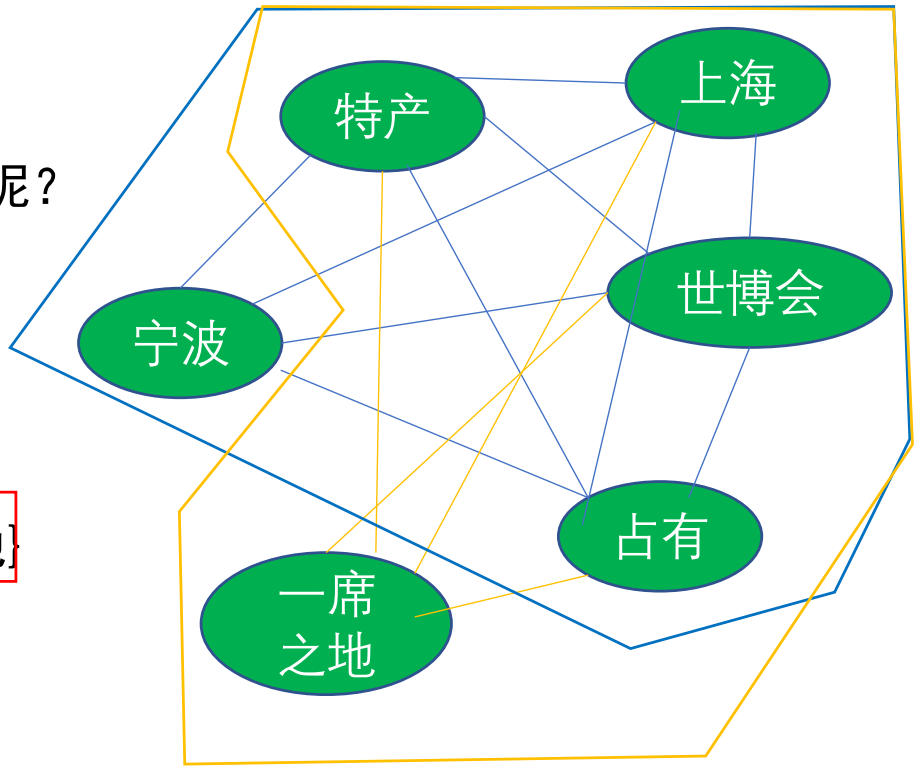
Keyword Extraction——Graph Construction

例如这个例句：

宁波有什么特产能在上海世博会占有一席之地呢？

设以下为分词结果，并设定滑动窗口大小为5

{宁波，特产，上海，世博会，占有，一席之地}



关键字提取——PageRank变种

Keyword Extraction——The Variation of PageRank

注意到原始的*PageRank*算法是使用在有向图上的，而现在我们尝试将该算法利用在无向图上。这么做的特点是由有向图所引起的*Page Leak*和*Page Sink*问题会得到改善。

首先依然要构建概率转移矩阵M，此时M应该是对称的；初始化*Rank*向量R，这里R中的每一个元素值初始化为1

然后利用迭代公式：

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

迭代至收敛即可，公式中的参数意义与*PageRank*相同。

此时可以将PR值从大到小进行排序，进而得到文章段落的关键词。

如果文本中存在若干个关键词相邻的情况，那么它们可以合并构建成关键短语。

关键句提取——基本思想

Key Sentence Extraction——Overview

将每一个句子看成图中的一个节点，若两个句子之间具有相似性，认为对应的两个节点之间有一个无向有权边，权值是相似度。

相似度计算公式如下：

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{\log(|S_i|) + \log(|S_j|)}$$

其中，S_i, S_j表示两个句子，w_k表示句中的词。

由于图变为了无向有权图，故对原始*PageRank*公式进行修改，其中w为权重，也就是相似度：

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

目录

实例应用——关键词提取

Experiment——Keyword Extraction

算法步骤总结：

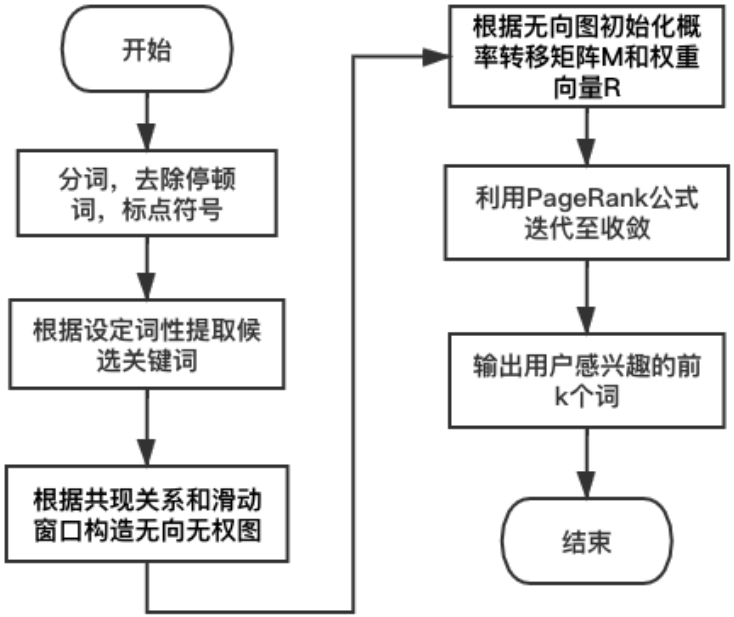
1、将文本整体当作一个句子进行处理。对文本按照词性进行分词，去除停顿词等无用词，去除标点符号。

2、将由（1）取得的候选关键词作为图的顶点，利用滑动窗口和共现机制来构造边。

3、初始化概率转移矩阵M和权重向量R

4、利用*PageRank*公式进行迭代至收敛

5、对（4）的结果进行降序排序，输出用户感兴趣的前k个词



*TextRank*提取关键词算法流程图

实例应用——关键词提取

Experiment——Keyword Extraction

考虑下面这样一段文本：

汽车驶进了科学城，两旁上世纪五六十年代的建筑在雪雾中掠过，有一次，我肯定看到了一尊列宁的塑像。这是一个让人产生怀旧感的城市，那些有上千年历史的古城并不能使人产生这种感觉，它们太旧了，旧得与你没有关系，旧得让人失去了感觉。但像这样年轻的城市，却使你想起一个刚刚逝去的时代，在那个时代你度过了童年和少年，那是你自己的上古时代，你自己的公元前。

我们利用分词工具提取其中的名词，如下：

{汽车，科学城，世纪，建筑，雪雾，列宁，塑像，怀旧感，城市，历史，古城，感情，关系，感觉，城市，时代}

假设滑动窗口的大小为2，是否可以直接利用候选关键词构造无向图？

文本节选自刘慈欣《球状闪电》上篇 西伯利亚

实例应用——关键词提取

Experiment——Keyword Extraction

答案是否定的。

这段文本更多的是讲述俄罗斯的科学城带给人的一种怀旧感，给予人人生的思考。

所以，如果对以上提取出的候选关键词分级，那么排在最前面的应该是“怀旧感”。但是，程序计算出来的排在第一的关键字却是“城市”，显然产生了偏差。它改变了文章结构。

正确的做法是在利用滑动窗口机制时，考虑所有词，在构建图时，仅考虑我们筛选出来的词。

```
使用论文的公式:
(1.7438741769883528, '城市')
(1.373114638985108, '感觉')
(1.1790422570827563, '科学城')
(1.1790422570827563, '列宁')
(1.119203356952331, '雪雾')
(1.119203356952331, '世纪')
(1.1012390534214294, '建筑')
```

实例应用——关键词提取

Experiment——Keyword Extraction

下表展示了*TextRank*算法在提取关键词方面的性能，实验是对于一篇摘要，由专业人士进行关键词抽取，再将各种关键词算法计算的结果与人工抽取结果相比较。

结果显示，在关键词的命中率（*Correct*）方面，比其他算法要优秀，同时，在召回率（*Recall*）的指标上根据有监督学习算法相比，也有良好的表现。

Method	Assigned		Correct		Precision	Recall	F-measure
	Total	Mean	Total	Mean			
TextRank							
Undirected, Co-occ.window=2	6,784	13.7	2,116	4.2	**31.2**	43.1	**36.2**
Undirected, Co-occ.window=3	6,715	13.4	1,897	3.8	28.2	38.6	32.6
Undirected, Co-occ.window=5	6,558	13.1	1,851	3.7	28.2	37.7	32.2
Undirected, Co-occ.window=10	6,570	13.1	1,846	3.7	28.1	37.6	32.2
Directed, forward, Co-occ.window=2	6,662	13.3	2,081	4.1	31.2	42.3	35.9
Directed, backward, Co-occ.window=2	6,636	13.3	2,082	4.1	31.2	42.3	35.9
Hulth (2003)							
Ngram with tag	7,815	15.6	1,973	3.9	25.2	**51.7**	33.9
NP-chunks with tag	4,788	9.6	1,421	2.8	29.7	37.2	33.0
Pattern with tag	7,012	14.0	1,523	3.1	21.7	39.9	28.1

Table 1: Results for automatic keyword extraction using TextRank or supervised learning (Hulth, 2003)

表格截至《*TextRank:Bringing Order into Texts*》

实例应用——关键词提取

Experiment——Keyword Extraction

Demo

实例应用——关键词提取

Experiment——Keyword Extraction

考虑以下文本，尝试人工提取其关键词

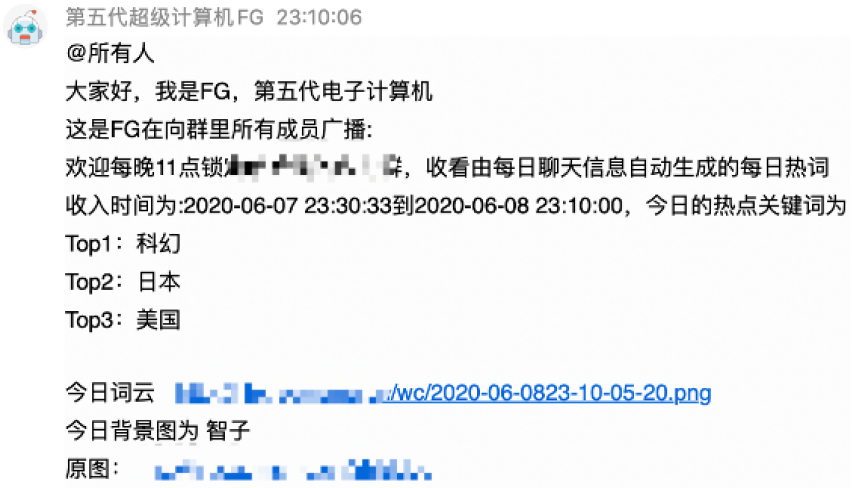
死星平静地燃烧了四亿八千万年，它的生命壮丽辉煌，但冷酷的能量守恒定律使它的内部不可避免地发生了一些变化：核火焰消耗着氢，而核聚变的产物氦，沉积到星体的中心并一点点地累积起来。这变化对于拥有巨量物质的死星来说是极其缓慢的，人类的整个历史对它来说不过是弹指一挥间。但四亿八千万年的消耗终于产生了它能感觉到的结果——惰性较大的氦已沉积到了相当的数量，它那曾是能量源泉的心脏渐渐变暗，死星老了。

文本节选自刘慈欣《超新星纪元》1991年版 死星

实例应用——FG

Application——FG

- *QQ*群机器人
- 基于*TextRank*算法+可视化技术提取每日聊天热点
- 分词使用*jieba*分词工具，并自定义用户字典
- 已在*GitHub*上开源10 *Stars*，2 *Forks*：
  https://github.com/mgsky1/FG



实例应用——FG

Application——FG

- 关键词提取有多种方式，最简单的为统计词频，但是这种方式效果差，不能很好反应文本内容。
- *FG*将*TextRank*算法应用到了*QQ*群聊天场景中，将每日的聊天记录当成文章提取每日的聊天主题。
- 结合文本可视化技术，将PR值当作词频传给标签云，进而达到凸显关键词大小轻重的效果。

目录

总结

Conclusion

- *TextRank*是一个优秀的无监督关键词/关键句提取算法，与其他监督型算法相比，不依赖语料。
- *TextRank*虽然是基于*Google*的*PageRank*演变而来，但是其他的一些基于图的分级算法也可以移植。
- *TextRank*算法中，对于每一个词的初始权重是一样的，对于大规模的文本来说，不具有针对性。

参考文献

References

[1]共现关系_tian_panda的博客-CSDN博客_共现关系[EB/OL]. [2020-06-20]. https://blog.csdn.net/tian_panda/article/details/81127034?utm_source=blogxgwz8.

[2]MIHALCEA R, TARAU P. TextRank: Bringing Order into Text.[C]//2004.

[3]张雯. TextRank算法的改进及在政法全文检索系统中的应用[D]. 广西大学, 2015.

[4]Textrank算法介绍 - 绽放的四叶草 - 博客园[EB/OL]. [2020-06-20]. https://www.cnblogs.com/clover-siyecao/p/5726480.html.

谢谢大家！