hanlp关键词提取TextRank算法

原创 阿童木 老实人阿童木 2019-12-09

TextRank是在Google的PageRank算法启发下，针对文本里的句子设计的权重算法，目标是自动摘要。它利用投票的原理，让每一个单词给它的邻居（术语称窗口）投赞成票，票的权重取决于自己的票数。这是一个“先有鸡还是先有蛋”的悖论，PageRank采用矩阵迭代收敛的方式解决了这个悖论。引用自http://www.hankcs.com/nlp/textrank-algorithm-to-extract-the-keywords-java-implementation.html。本博文通过hanlp关键词提取的一个Demo，并通过图解的方式来讲解TextRank的算法。

```
//长句子
        String content = "程序员(英文Programmer)是从事程序开发、维护的专业人员。" +
                "一般将程序员分为程序设计人员和程序编码人员，" +
                "但两者的界限并不非常清楚，特别是在中国。" +
                "软件从业人员分为初级程序员、高级程序员、系统" +
                "分析员和项目经理四大类。";
```

最后提取的关键词是：[程序员, 程序, 分为, 人员, 软件]

下面来分析为什么会提取出这 5 个关键词

第一步：分词

把content 通过一个的分词算法进行分词，这里采用的是Viterbi算法也就是HMM算法，具体请参与我的另篇文章https://blog.csdn.net/zhaojianting/article/details/78194317。分词后（当然首先应把停用词、标点、副词之类的去除）的结果是：

[程序员, 英文, Programmer, 从事, 程序, 开发, 维护, 专业, 人员, 程序员, 分为, 程序, 设计, 人员, 程序, 编码, 人员, 界限, 并不, 非常, 清楚, 特别是在, 中国, 软件, 从业人员, 分为, 程序员, 高级, 程序员, 系统分析员, 项目经理, 四大]

第二步：构造窗口

hanlp的实现代码如下：

```
Map<String, Set<String>> words = new TreeMap<String, Set<String>>();
        Queue<String> que = new LinkedList<String>();
        for (String w : wordList)
        {
            if (!words.containsKey(w))
            {
                words.put(w, new TreeSet<String>());
            }
            // 复杂度O(n-1)
            if (que.size() >= 5)
            {
                que.poll();
            }
            for (String qWord : que)
            {
                if (w.equals(qWord))
                {
                    continue;
                }
                //既然是邻居,那么关系是相互的,遍历一遍即可
                words.get(w).add(qWord);
                words.get(qWord).add(w);
            }
            que.offer(w);
        }
```

这个代码的功能是为分个词构造窗口，这个词前后各四个词就是这个词的窗口，如词分词后一个词出现了多次，像**[程序员]**，那就是把每次出现取一次窗口，然后把各次结果合并去重，最后结果是：**程序员=[Programmer, 专业, 中国, 人员, 从业人员, 从事, 分为, 四大, 开发, 程序, 系统分析员, 维护, 英文, 设计, 软件, 项目经理, 高级]**。最后形成的窗口：

```
Map<String, Set<String>> words  =

{Programmer=[从事, 开发, 程序, 程序员, 维护, 英文], 专业=[人员, 从事, 分为, 开发, 程
```

第三步：迭代投票

每个词最后的投票得分由这个词的窗口进行多次迭代投票决定，迭代的结束条件就是大于最大迭代次数这里是200次，或者两轮之前某个词的权重小于某一值这里是0.001f。看下代码：

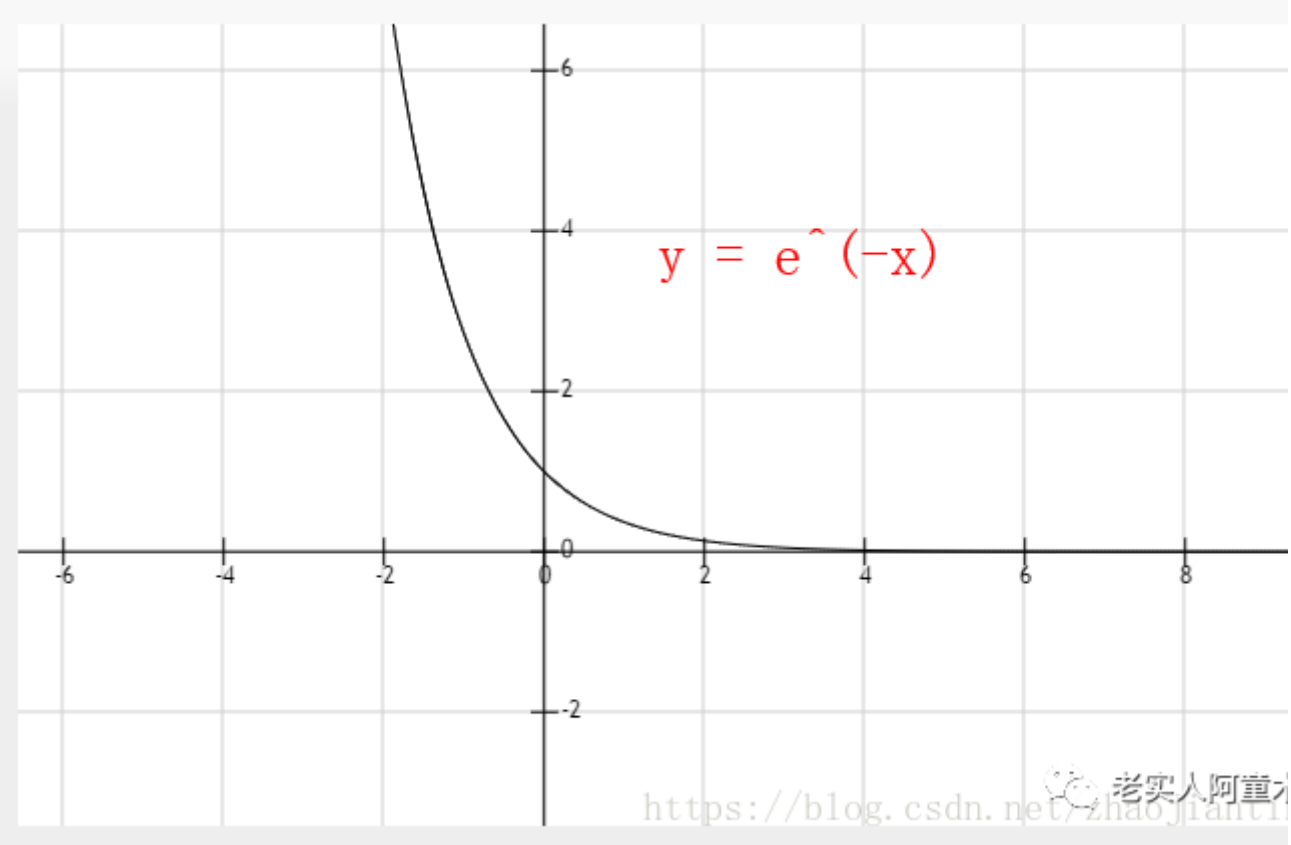
```
Map<String, Float> score = new HashMap<String, Float>();
        //依据TF来设置初值
        for (Map.Entry<String, Set<String>> entry : words.entrySet()){
          score.put(entry.getKey(),sigMoid(entry.getValue().size()));
        }
        System.out.println(score);
        for (int i = 0; i < max_iter; ++i)
        {
            Map<String, Float> m = new HashMap<String, Float>();
            float max_diff = 0;
            for (Map.Entry<String, Set<String>> entry : words.entrySet())
            {
                String key = entry.getKey();
                Set<String> value = entry.getValue();
                m.put(key, 1 - d);
                for (String element : value)
                {
                    int size = words.get(element).size();
                    if (key.equals(element) || size == 0) continue;
                    m.put(key, m.get(key) + d / size * (score.get(element) ==
                }
                max_diff = Math.max(max_diff, Math.abs(m.get(key) - (score.ge
            }
            score = m;
            if (max_diff <= min_diff) break;
        }

        System.out.println(score);
        return score;
    }
```

投票的原理拿Programmer=**[从事, 开发, 程序, 程序员, 维护, 英文]**，这个词来说明，Programmer最后的得分是由**[从事, 开发, 程序, 程序员, 维护, 英文]**，这 6 个词依次投票决定的，每个词投出去的分数是和他本身的权重相关的。

投票开始前每个词初始化了一个权重，score.put(entry.getKey(),sigMoid(entry.getValue().size()))，这个权重是0到1之间，公式是

```
//value是每个词窗口的大小
    public static float sigMoid(float value) {
      return (float)(1d/(1d+Math.exp(-value)));
    }
```

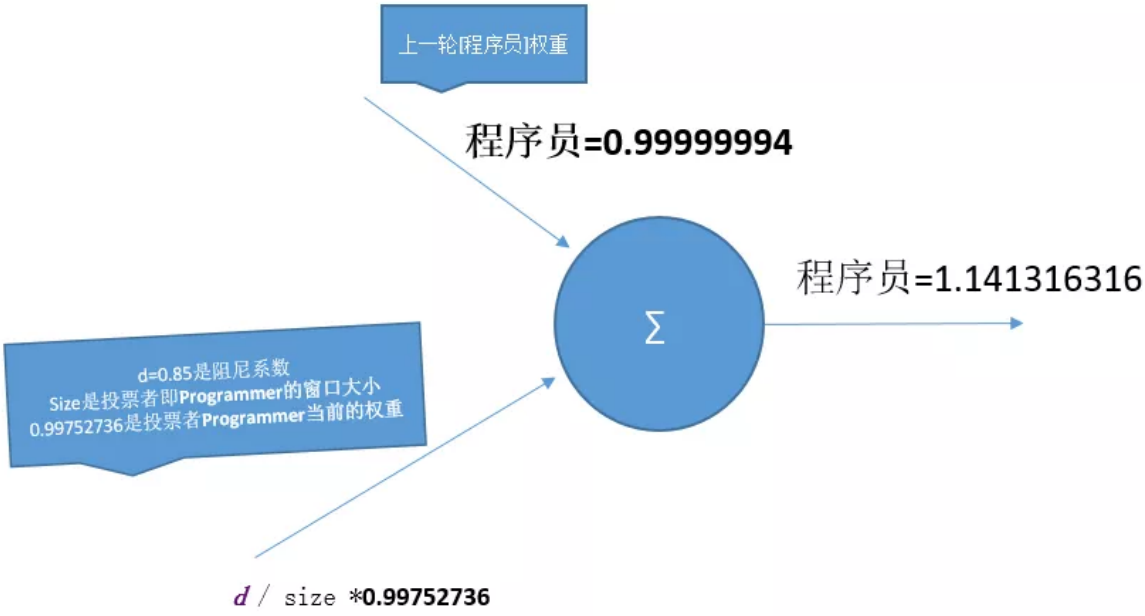
这个函数的公式和图像如下,因为value一定是大于0的，所以sigMod值属于(0,1)：



初始化后的分词是：**{特别是在=0.99966466，程序员=0.99999994，编码=0.99752736，四大=0.98201376，英文=0.9933072，非常=0.99966466，界限=0.99908894，系统分析员=0.9933072，从业人员=0.99908894，程序=0.99999774，专业=0.99908894，项目经理=0.98201376，设计=0.9933072，从事=0.99908894，Programmer=0.99752736，软件=0.99966466，人员=0.99999386，清楚=0.99966466，中国=0.99966466，开发=0.99966466，并不=0.99966466，高级=0.99908894，分为=0.99999386，维护=0.99966466}**

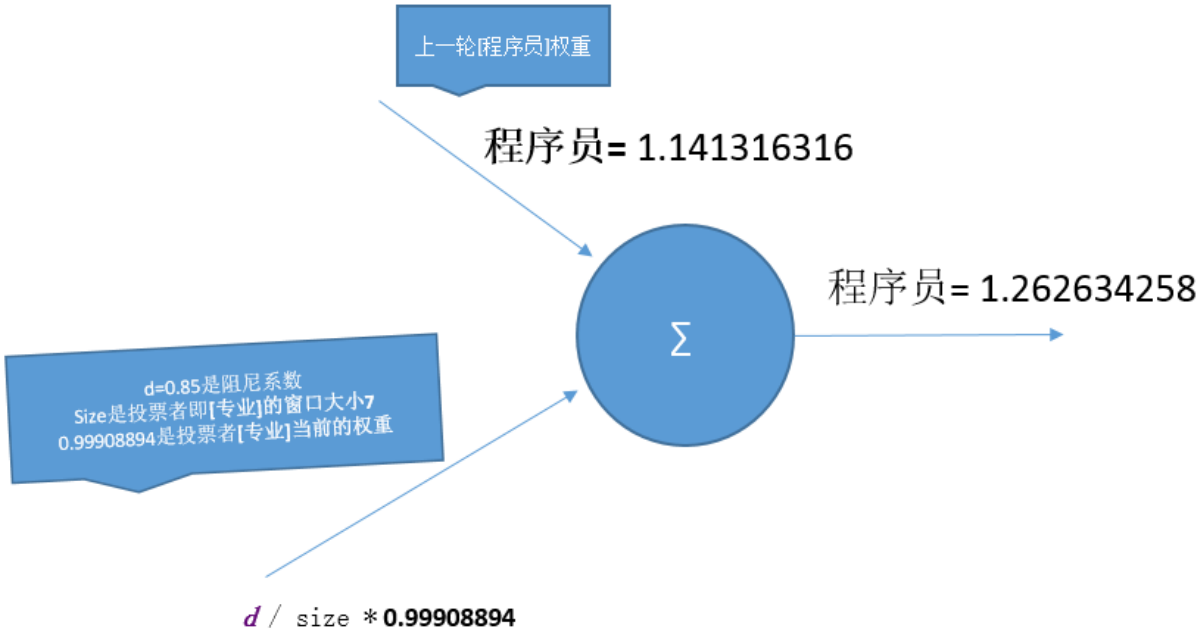
进行迭代投票，第一轮投票，**[Programmer，专业，中国，人员，从业人员，从事，分为，四大，开发，程序，系统分析员，维护，英文，设计，软件，项目经理，高级]**依次给**程序员**投票，得分如下：

[Programmer]给**[程序员]**投票后，**[程序员]**的得分:



老实人阿童木
https://blog.csdn.net/zhaojianting

[专业]给**[程序员]**投票:



老实人阿童木
https://blog.csdn.net/zhaojianting

这样**[Programmer, 专业, 中国, 人员, 从业人员, 从事, 分为, 四大, 开发, 程序, 系统分析员, 维护, 英文, 设计, 软件, 项目经理, 高级]**依次给**[程序员]**投票，投完票后，再

给其它的词进行投票，本轮结束后，判断是否达到最大迭代次数200或两轮之间分数差值小于0.001，如果满足则结束，否则继续进行迭代。

最后的投票得分是：{特别是在=1.0015739, 程序员=2.0620303, 编码=0.78676623, 四大=0.6312981, 英文=0.6835063, 非常=1.0018439, 界限=0.88890904, 系统分析员=0.74232763, 从业人员=0.8993066, 程序=1.554001, 专业=0.88107216, 项目经理=0.6312981, 设计=0.6702926, 从事=0.9027207, Programmer=0.7930236, 软件=1.0078223, 人员=1.4288887, 清楚=0.9998723, 中国=0.99726284, 开发=1.0065585, 并不=0.9968608, 高级=0.9673803, 分为=1.4548829, 维护=0.9946941}，分数最高的关键词就是要提取的关键词。

作者：阿童木，脚踏大数据、商业智能、人工智能、物联网，四座高峰，易学大师！