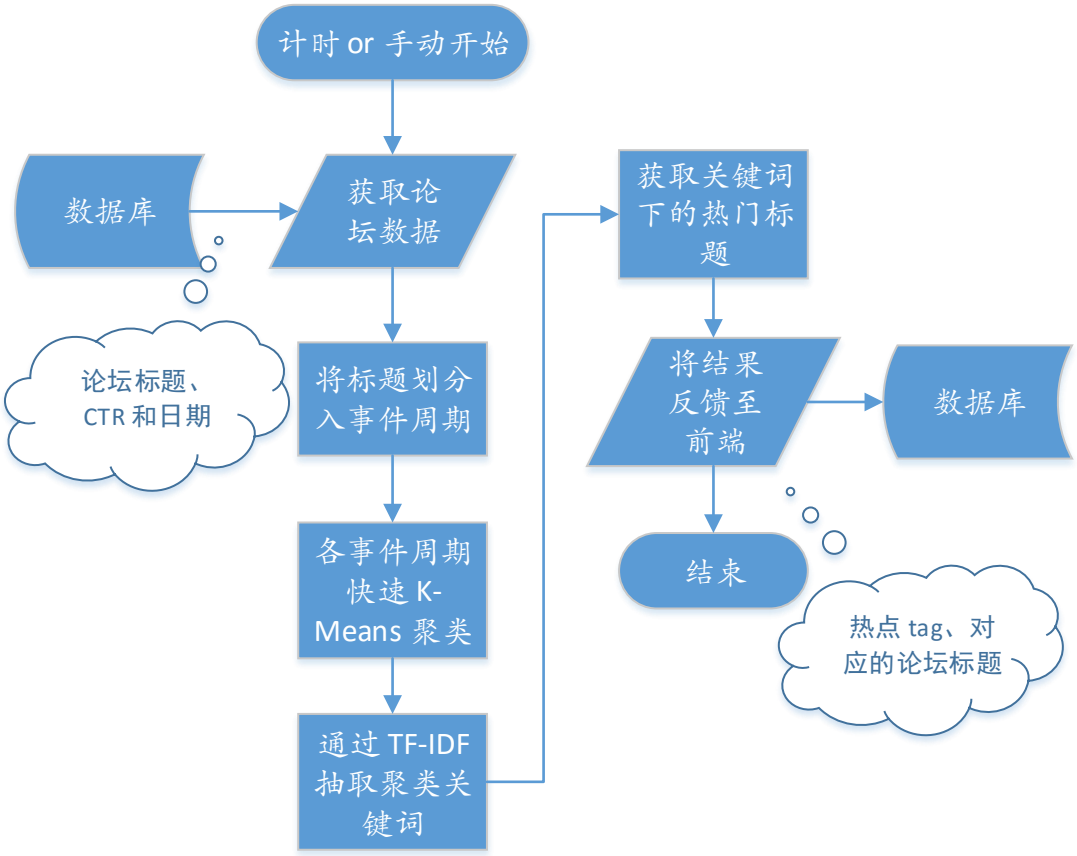
# 舆情挖掘算法简述

李润东 2016/5/21

在这次构建基于丁香园论坛标题信息及点击量的舆情挖掘系统中，我主要负责舆情挖掘算法的设计和实施。现对整个算法的框架和技术细节进行一个简要的说明。

## 一、 算法的执行框架




## 二、 各部分技术原理

### 1. 输入数据格式

输入数据为爬虫从丁香园论坛得到的论坛帖子信息，包括：帖子标题 (title)、链接 (URL)、发帖日期 (time) 和当前点击量 (CTR)。为下文描述方便考虑，将每一条记录记为

$$\boldsymbol{X}_i = \{title_i,\ URL_i, time_i, CTR_i\}$$

其中，$i$ 表示帖子记录的索引 (index)。数据采用 UTF-8 编码。

2. 事件周期划分

按照新闻热点生命周期普遍为 14 天的原则，对所有输入的 $\boldsymbol{X}_i$ 按照 $time_i$ 进行分类，将其归入不同的热点周期 $Per_j$ 中，其中 $j$ 表示该热点周期的索引。在新闻热点生命周期（记为 $hot_{const}$，默认值为 14 ）可自定义的情况下，对于某一 $\boldsymbol{X}_i$，其所属热点周期 $j$ 的计算方式可表示为：

$$j_{begin} = (time_i - time_0) - (hot_{const} - 1)$$
$$j_{end} = time_i - time_0$$
$$\boldsymbol{X}_i[Per] = [j_{begin}\ \ to\ \ j_{end}]$$

其中， $time_0$ 表示所有发帖记录中最早的一天，并记最大热点周期编号为 $J$ 。可以看到，每个帖子可能属于多个热点周期，而每个热点周期内又可能有不同的关键词。

3. 各周期数据聚类

为确定每个热点周期内都发生了那些事情，我们需要对每个热点周期内的所有标题做聚类处理。然而，聚类是一种数值方法，为完成聚类，我们需要把热点周期 $j \in [0, J]$ 中的各 $\boldsymbol{X}_i$ 根据其 $title_i$，将其映射到向量空间 $\boldsymbol{U}_j^k = \{u_1, u_2, \dots, u_k\}$ 中。具体方法如下：

```
For j ∈ [0 : J]:
    /* 将属于热点 j 的所有标题文字连在一起进行分词，得到热点 j 下的词汇表 */
    Token_List = [].join(title_i.split() For i ∈ j.index)
    For i ∈ j.index: /* 循环热点 j 下的所有发帖记录 */
        X_i[oneHot] = [token in title_i.split() For token in Token_List]   /* 根据 title_i 中是否包含 Token_List 中的词汇，对每一个帖子题目进行独热码编码 */
        /* 开始进行聚类处理 */
        ……
```

可以看到，上文中的 K=length(Token_List)。对于每一不同的热点 j，由于其包含的帖子题目不一样，故其向量空间 $\boldsymbol{U}_j^k$ 也不一样。在不同的 $\boldsymbol{U}_j^k$ 中，即使是相同的 $title_i$ 也会得到不同的独热码映射，最终会把每一条发帖记录映射为类似 $< 0, 1, 1, 0, \dots >$ 这样的形式。之后就可以进行聚类处理了。

考虑到 $\boldsymbol{U}_j^k$ 维度较高，这里我们使用了 Mini-Batch K-Means 聚类算法。该算法与传统的 K-Means 算法近似，输入聚类数目 k ，并将热点 j 下的发帖记录 $X_i$ 划分至不重叠的聚类 $C_l, l \in [1, k]$ 中；对于 $X_i \in C_l$，通过不断迭代 $C_l$ 的中心点 $\mu_i$，使得目标函数

$$\sum_{X_i \in C_l} \min(\|X_i[one_{hot}] - \mu_i\|^2)$$

最小。由于 $\|\cdot\|$ 算符中所有运算数均已编码为独热码形式，故直接使用欧氏距离表示 $\|\cdot\|$。

Mini-Batch K-Means 算法比 K-Means 算法收敛速度快很多，原因是在算法启动时，除了随机挑选 $\mu_i$ ，还随机选取 b 个点，分别作为离其最近的 $\mu_i$ 的 mini-batch；在迭代 $\mu_i$ 时，各 $\mu_i$ 从其 mini-batch 中选取下一个点作为新的 $\mu_i$，比起从 $C_l$ 中迭代下一个中心点，这样无疑大大减少了候选点的数量，而且结果和 K-Means 几乎没有区别。该算法的具体实现细节可参考 [Sculley, 2010]。

4. 各聚类关键词抽取

在对各热点周期内的帖子标题聚类结束后，可以在 j 个热点周期下，各得出 k 个聚类。下面，我们只要提取出每个聚类的关键词，就可方便地进行结果展示了。这里我们将每个聚类下的所有帖子标题连接起来，再使用 TF-IDF （词频-逆文档频率）评估该字符串中的每个单词，从中抽取出 3 个关键词（关键词数目可以由用户指定）。

关于 TF-IDF 的计算方法如下：对于聚类 k 中所有标题连成的文档 $d_k$，将其做分词处理后，对于每一个单词 $t_j$，TF-IDF($d_k$, $t_j$) 的计算可表示为

$$\mathrm{TF-IDF}(d_k, t_j) = \mathrm{TF}(d_k, t_j) \cdot log\frac{N}{n_j}$$

其中 $\mathrm{TF}(d_k, t_j)$ 表示单词 $t_j$ 在文档 $d_k$ 中出现的词频，N 为本次计算中文档的个数（此处 N = j * k），$n_j$ 表示包含单词 $t_j$ 的文档个数。TF-IDF 的直觉解释是：在一篇文档中频繁出现（TF 很大）但很少出现在其他文档中的单词，与该文档主题相关的可能性很大。[Ricci, et al. 2015]

5. 各关键词热点标题抽取

这部分就非常简单了，直接在每个事件中的每个关键词下，点击量最高的前 3 个（这里也可以自定义）帖子标记出来就行。

## 三、 各部分实现方式

算法整体使用 Python 编程实现。由于算法各个子部分的各自算法都已经相当成熟，故在实现过程中大量使用了现有的软件包。具体如下：

- 分词： `jieba.lcut`
- Mini-Batch K-Means 聚类： `sklearn.cluster.MiniBatchKMeans`
- TF-IDF 关键词抽取： `jieba.analyse.extract_tags`

其他边角处理，例如数据预处理、结果封装，则直接自己编程完成。上层的框架由曾智师同学负责实现，我不太清楚具体细节。

## 四、 关于进一步改进

下面的改进目标是，实现各个热点周期内是否的确有热点事件发生的判断（正在实现），标题权重化处理（正在实现），分词时近义词同义词的处理，以及标题语义化处理。

## 五、 参考文献


[1]. Sculley D. Web-scale k-means clustering[C]//Proceedings of the 19th international conference on World wide web. ACM, 2010: 1177-1178.

[2]. F. Ricci, et al. Recommender Systems Handbook. Springer. 2015: 56.