Poll的笔记

- [三叶草精神] what hurts mor
pain of hard work or the pai
regret?

首页
管理 博

随笔 - 63 文章 - 1 评论

StatCounter - Free Web Tr
er and Counter

昵称：Poll的笔记
园龄：3年
粉丝：1056
关注：14
+加关注

‹ 2018年6月

日	一	二	三	四	五
27	28	29	30	31	1
3	4	5	6	7	8
10	11	12	13	14	15
17	18	19	20	21	22
24	25	26	27	28	29
1	2	3	4	5	6

最新随笔

1. [Machine Learning] 深度学习中的梯度
2. [Machine Learning] logistic函oftmax函数
3. [Machine Learning & Algorith经网络基础
4. [Machine Learning] Active Leg
5. [Machine Learning & Algorith ML机器学习系列2：深入浅出ML ropy-Based家族
6. [Machine Learning & Algorith ML机器学习系列1：深入浅出ML ression家族
7. [Data Structure] LCSs——最共子序列和最长公共子串
8. [Algorithm & NLP] 文本深度表型——word2vec&doc2vec词向量
9. [Algorithm] 机器学习算法常用总结
10. [Linux] Linux常用文本操作命理

[Algorithm] 使用SimHash进行海量文本去重

阅读目录

在之前的两篇博文分别介绍了常用的hash方法（[Data Structure & Algorithm] Hash那点事儿）以及局部敏感hash算法（[Algorithm] 局部敏感哈希算法(Locality Sensitive Hashing)），本文介绍的SimHash是一种局部敏感hash，它也是Google公司进行海量网页去重使用的主要算法。

回到顶部

1. SimHash与传统hash函数的区别

传统的Hash算法只负责将原始内容尽量均匀随机地映射为一个签名值，原理上仅相当于伪随机数产生算法。传统的hash算法产生的两个签名，如果原始内容在一定概率下是相等的；如果不相等，除了说明原始内容不相等外，不再提供任何信息，因为即使原始内容只相差一个字节，所产生的签名也很可能差别很大。所以传统的Hash是无法在签名的维度上来衡量原内容的相似度，而SimHash本身属于一种局部敏感哈希算法，它产生的hash签名在一定程度上可以表征原内容的相似度。

我们主要解决的是文本相似度计算，要比较的是两个文章是否相识，当然我们降维生成了hash签名也是用于这个目的。看到这里估计大家就明白了，我们使用的simhash就算把文章中的字符串变成 01 串也还是可以用于计算相似度的，而传统的hash却不行。我们可以来做个测试，两个相差只有一个字符的文本串，“你妈妈喊你回家吃饭哦，回家罗回家罗”和“你妈妈叫你回家吃饭啦，回家罗回家罗”。

通过simhash计算结果为：

1000010010101101111111000001010110100010011111000010101100101011

1000010010101101011111100000101011010001001111100001101010001011

通过传统hash计算为：

00010000011001101001110110111 10

10100100011111111100101100111 01

大家可以看得出来，相似的文本只有部分 01 串变化了，而普通的hash却不能做到，这个就是局部敏感哈希的魅力。

回到顶部

2. SimHash算法思想

8 0

假设我们有海量的文本数据，我们需要根据文本内容
对于文本去重而言，目前有很多NLP相关的算法可以在很

随笔分类

Algorithm(23)

Bash(1)

C/C++(6)

Computational Advertising(1)

Data Structure(6)

Database(3)

Evolutionary Algorithm(2)

Hadoop(4)

Linux(6)

Machine Learning(15)

Math(2)

Network(2)

Operate System

Python(11)

Recommendation System(1)

Search Engine(3)

Social Network Analysis(1)

Web Development(2)

生活杂谈(1)

随笔档案

2017年1月 (1)

2016年7月 (1)

2016年6月 (1)

2016年5月 (4)

2016年4月 (2)

2016年3月 (2)

2016年2月 (2)

2016年1月 (1)

2015年12月 (5)

2015年11月 (3)

2015年10月 (1)

2015年9月 (5)

2015年8月 (8)

2015年7月 (8)

2015年6月 (19)

My Team

OMEGA team

但是我们现在处理的是大数据维度上的文本去重，这就对算法的效率有着很高的要求。而局部敏感hash算法可以将原始的文本内容映射为数字（hash签名），而且较为相近的文本内容对应的hash签名也比较相近。SimHash算法是Google公司进行海量网页去重的高效算法，它通过将原始的文本映射为64位的二进制数字串，然后通过比较二进制数字串的差异进而来表示原始文本内容的差异。

回到顶部

3. SimHash流程实现

simhash是由 Charikar 在2002年提出来的，本文为了便于理解尽量不使用数学公式，分为这几步：

（注：具体的事例摘自Lanceyan的博客《海量数据相似度计算之simhash和海明距离》）

- **1、分词**，把需要判断文本分词形成这个文章的特征单词。最后形成去掉噪音词的单词序列并为每个词加上权重，我们假设权重分为5个级别（1~5）。比如：“ 美国“51区”雇员称内部有9架飞碟，曾看见灰色外星人 ” ==> 分词后为 “ 美国（4） 51区（5） 雇员（3） 称（1） 内部（2） 有（1） 9架（3） 飞碟（5） 曾（1） 看见（3） 灰色（4） 外星人（5）”，括号里是代表单词在整个句子里重要程度，数字越大越重要。
- **2、hash**，通过hash算法把每个词变成hash值，比如“美国”通过hash算法计算为 100101,“51区”通过hash算法计算为 101011。这样我们的字符串就变成了一串串数字，还记得文章开头说过的吗，要把文章变为数字计算才能提高相似度计算性能，现在是降维过程进行时。
- **3、加权**，通过 2步骤的hash生成结果，需要按照单词的权重形成加权数字串，比如“美国”的hash值为“100101”，通过加权计算为“4 -4 -4 4 -4 4”；“51区”的hash值为“101011”，通过加权计算为 “ 5 -5 5 -5 5 5”。
- **4、合并**，把上面各个单词算出来的序列值累加，变成只有一个序列串。比如 “美国”的 “4 -4 -4 4 -4 4”，“51区”的 “ 5 -5 5 -5 5 5”， 把每一位进行累加， “4+5 -4+-5 -4+5 4+-5 -4+5 4+5” ==》 “9 -9 1 -1 1 9”。这里作为示例只算了两个单词的，真实计算需要把所有单词的序列串累加。
- **5、降维**，把4步算出来的 “9 -9 1 -1 1 9” 变成 0 1 串，形成我们最终的simhash签名。 如果每一位大于0 记为 1，小于0 记为 0。最后算出结果为：“1 0 1 0 1 1”。

整个过程的流程图为：

常用链接

[Andrew Moore] Statistical Dating Tutorials

[Online Terminals] tutorialspoin

ACM之家

机器学习周报

开源中国

漫谈机器学习算法

鸟哥的Linux私房菜

统计之都

推酷

我爱公开课

我爱机器学习

我爱自然语言处理

推荐博友

CAML

计算广告与机器学习 - 技术共享平

Dustinsea

百度关键词搜索推荐系统maker

JasonDing

机器学习、算法、Spark

July的博客

结构之法，算法之道。

uc技术博客

UC企业技术博客

Vamei

文艺地讲解编程、数学和设计

阿哈磊

图文并茂的阿哈磊算法讲解，简单

董的博客

关注大规模数据处理

寒江独钓

详细的数据结构和算法讲解

火光摇曳

机器学习、分布式计算、计算广告

静觅

python爬虫系列教程

静逸

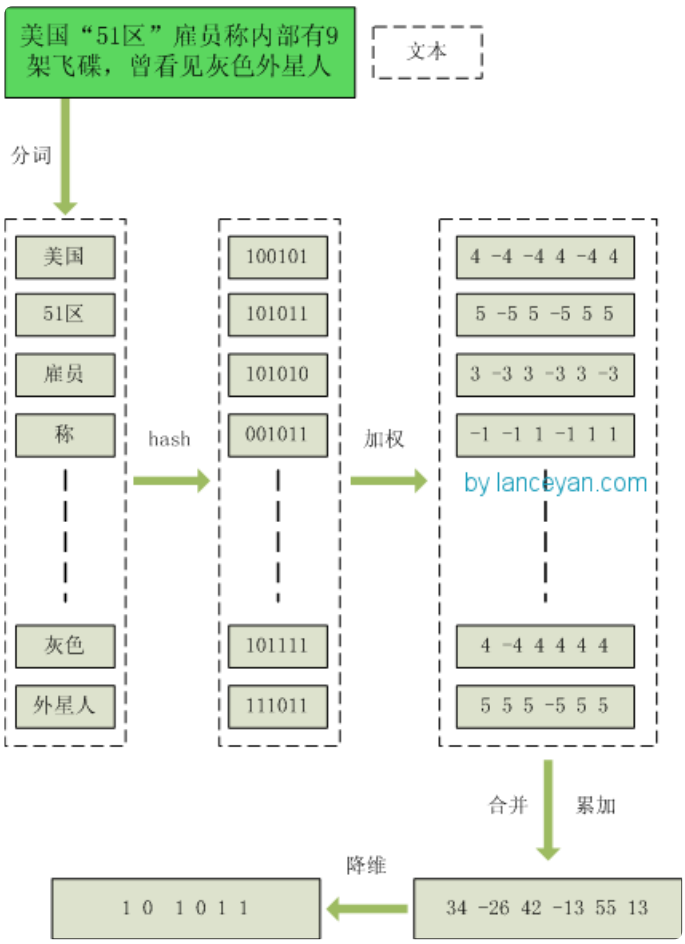
专注于wed前端

酷壳

程序员必看，涉及面很广，也很有

牛吧大数据

大数据、机器学习、R语言



回到顶部

4. SimHash签名距离计算

我们把库里的文本都转换为simhash签名，并转换为long类型存储，空间大大减少。现在我们虽然解决了空间，但是如何计算两个simhash的相似度呢？难道是比较两个simhash的01有多少个不同吗？对的，其实也就是这样，我们通过海明距离（Hamming distance）就可以计算出两个simhash到底相似不相似。两个simhash对应二进制（01串）取值不同的数量称为这两个simhash的海明距离。举例如下： **10101** 和 **00110** 从第一位开始依次有第一位、第四、第五位不同，则海明距离为3。对于二进制字符串的a和b，海明距离为等于在a XOR b运算结果中1的个数（普遍算法）。

回到顶部

5. SimHash存储和索引

经过simhash映射以后，我们得到了每个文本内容对应的simhash签名，而且也确定了利用汉明距离来进行相似度的衡量。那剩下的工作就是两两计算我们得到的simhash签名的汉明距离了，这在理论上是完全没问题的，但是考虑到我们的数据是海量的这一特点，我们是否应该考虑使用一些更具效率的存储呢？其实SimHash算法输出的simhash签名可以为我们很好建立索引，从而大大减少索引的时间，那到底怎么实现呢？

这时候大家有没有想到hashmap呢，一种理论上具有O(1)复杂度的查找数据结构。我们要查找一个key值时，通过传入一个key就可以很快的返回一个value，这个号称查找速度最快的数据结构是如何实现的呢？看下hashmap的内部结构：

阮一峰的网络日志

算法，数学，文学，科技，创业...

石山园

Hadoop入门进阶课程系列

淘宝技术部

淘宝技术介绍

王路情

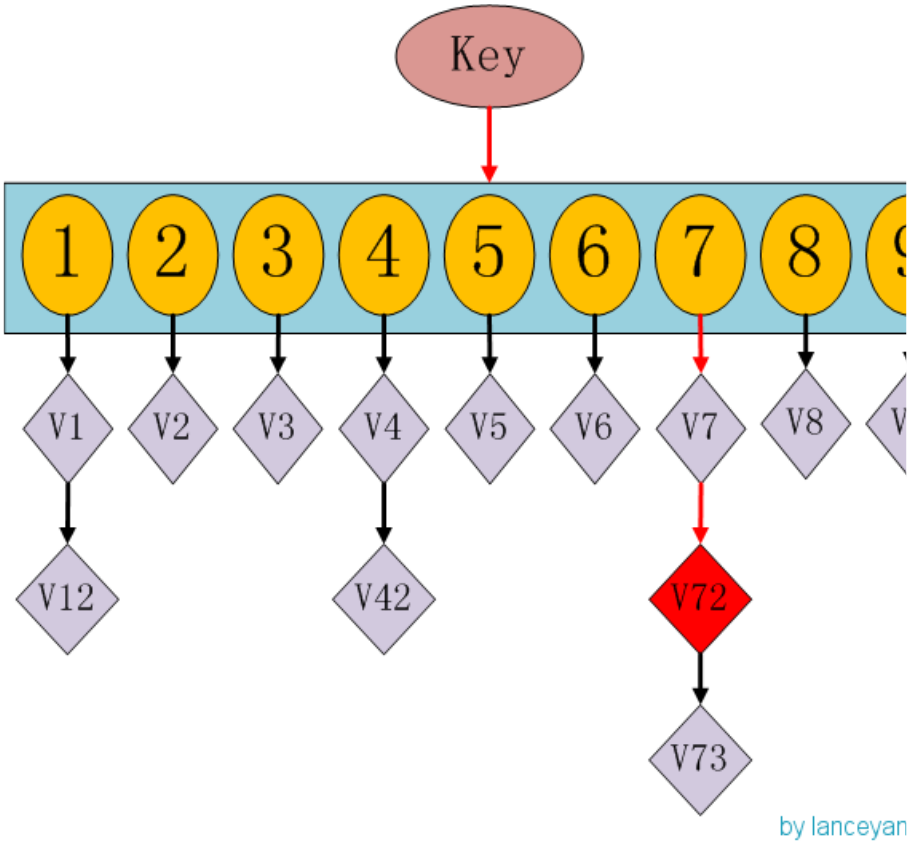
Hadoop研究和R实战

小坦克

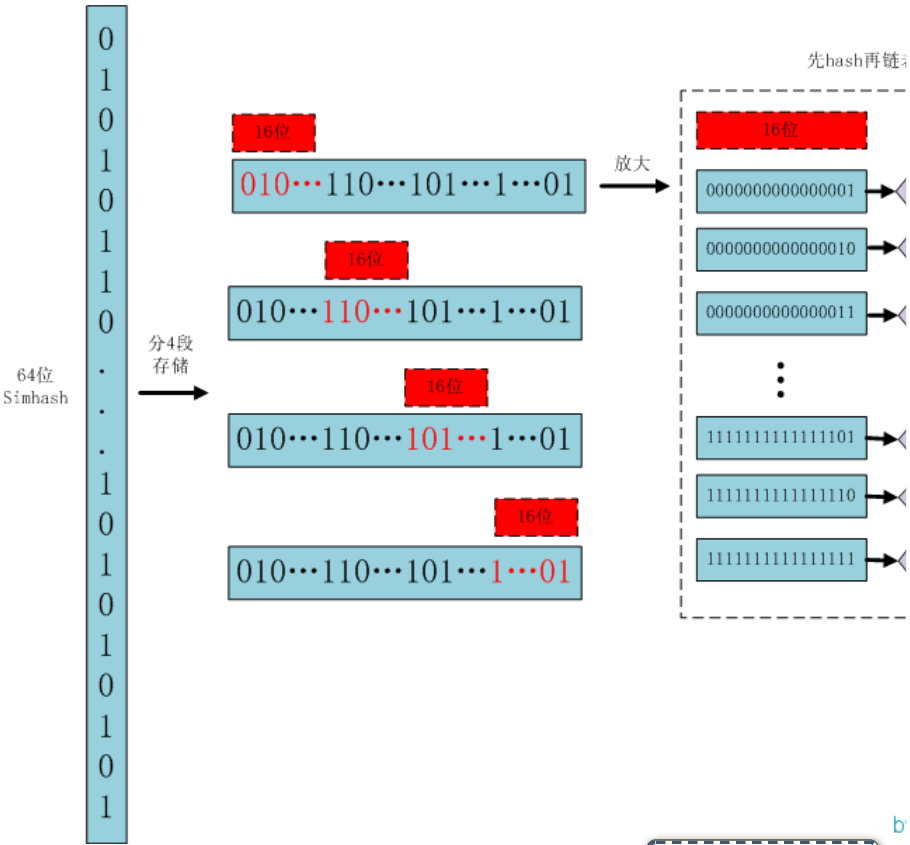
网络协议介绍

积分与排名

积分 - 161631
排名 - 1734



如果我们需要得到key对应的value，需要经过这些计算，传入key，计算key的hashcode，得到7的位置；发现7位置对应的value还有好几个，就通过链表查找，直到找到v72。其实通过这么分析，如果我们的hashcode设置的不够好，hashmap的效率也不见得高。借鉴这个算法，来设计我们的simhash查找。通过顺序查找肯定是不行的，能否像hashmap一样先通过键值对的方式减少顺序比较的次数。看下图：



存储：

1、将一个64位的simhash签名拆分成4个16位的二进
的16位）

2、分别拿着4个16位二进制码查找当前对应位置上是否有元素。（放大

后的16位）

3、对应位置没有元素，直接追加到链表上；对应位置有则直接追加到链表尾端。（图上的 S1 — SN）

查找：

1、将需要比较的simhash签名拆分成4个16位的二进制码。

2、分别拿着4个16位二进制码每一个去查找simhash集合对应位置上是否有元素。

3、如果有元素，则把链表拿出来顺序查找比较，直到simhash小于一定大小的值，整个过程完成。

原理：

借鉴hashmap算法找出可以hash的key值，因为我们使用的simhash是局部敏感哈希，这个算法的特点是只要相似的字符串只有个别的位数是有差别变化。那这样我们可以推断两个相似的文本，至少有16位的simhash是一样的。具体选择16位、8位、4位，大家根据自己的数据测试选择，虽然比较的位数越小越精准，但是空间会变大。分为4个16位段的存储空间是单独simhash存储空间的4倍。之前算出5000w数据是 382 Mb，扩大4倍1.5G左右，还可以接受

回到顶部

6. SimHash存储和索引

1. 当文本内容较长时，使用SimHash准确率很高，SimHash处理短文本内容准确率往往不能得到保证；

2. 文本内容中每个term对应的权重如何确定要根据实际的项目需求，一般是可以使用IDF权重来进行计算。

回到顶部

7. 参考内容

1. 严澜的博客《海量数据相似度计算之simhash短文本查找》

2. 《Similarity estimation techniques from rounding algorithms》

分类: Algorithm

标签: Algorithm

« 上一篇：[Hadoop] Hadoop学习历程 [持续更新中...]
» 下一篇：[C/C++] C/C++延伸学习系列之STL及Boost库概述

posted @ 2016-02-20 14:07 Poll的笔记 阅读(9862) 评论(7) 编辑 收藏

评论列表

#1楼 2016-04-02 20:13 dodo2016

赞一个

支持(0) 反对(0)

#2楼 2016-07-11 12:05 攻城小狮

总结的非常清晰，做了个很好的回顾。赞！

支持(0) 反对(0)

#3楼[楼主] 2016-07-11 12:06 Poll的笔记

@ 攻城小狮
多谢支持，希望能有帮助。

支持(0) 反对(0)

#4楼 2016-07-12 10:25 酱油羊

很好的文章，赞一个！关于第5节的索引我理解是一个 trieTree 和 hashMap 的 tradeof f。不知道对不对。

支持(0) 反对(0)

#5楼[楼主] 2016-07-12 11:28 Poll的笔记

@ 酱油羊
可以这样理解，为了实现快速分桶

支持(0) 反对(0)

#6楼 2016-08-01 18:00 千里缘

你好，我想请问下，如果进过simhash后，索引值分布不平均怎么办，我的400w文档索引值大部分都分布在0000000000000000，就是16位全是0这个。这样的数据大概有100万，也就是说索引基本没用，这种情况的话应该怎么做呢。

支持(0) 反对(0)

#7楼 2016-09-27 13:47 编程笔记

简洁明了的介绍

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。