

关闭旧主题浏览模式

互联网时代的社会语言学：基于SNS的文本数据挖掘

今年上半年，我在人人网实习了一段时间，期间得到了很多宝贵的数据，并做了一些还算有意义的事情，在这里和大家一块儿分享。感谢人人网提供的数据与工作环境，感谢赵继承博士、詹卫东老师的支持和建议。在这项工作中，我得到了很多与众人交流的机会，特别感谢 OpenParty、TEDxBeijing 提供的平台。本文已发表在了《程序员》杂志，分上下两部分刊于 2012 年 7 月刊和 8 月刊，在此感谢卢鹤翔编辑的辛勤工作。由于众所周知的原因，《程序员》刊出的文章被和谐过（看到后面大家就自动地知道被和谐的内容是什么了），因而我决定把完整版发在 Blog 上，同时与更多的人一同分享。对此感兴趣的朋友可以给我发邮件继续交流。好了，开始说正文吧。

作为中文系应用语言学专业的学生以及一名数学 Geek，我非常热衷于用计算的方法去分析汉语资料。汉语是一种独特而神奇的语言。对汉语资料进行自然语言处理时，我们会遇到很多其他语言不会有的困难，比如分词——汉语的词与词之间没有空格，那计算机怎么才知道，“已结婚的和尚未结婚的青年都要实行计划生育”究竟说的是“已 / 结婚 / 的 / 和 / 尚未 / 结婚 / 的 / 青年”，还是“已 / 结婚 / 的 / 和尚 / 未 / 结婚 / 的 / 青年”呢？这就是所谓的分词歧义难题。不过，现在很多语言模型已经能够比较漂亮地解决这一问题了。但在中文分词领域里，还有一个比分词歧义更令人头疼的东西——未登录词。中文没有首字母大写，专名号也被取消了，这叫计算机如何辨认人名地名之类的东西？更惨的则是机构名、品牌名、专业名词、缩略语、网络新词等等，它们的产生机制似乎完全无规律可寻。最近十年来，中文分词领域都在集中攻克这一难关。自动发现新词成为了关键的环节。

挖掘新词的传统方法是，先对文本进行分词，然后猜测未能成功匹配的剩余片段就是新词。这似乎陷入了一个怪圈：分词的准确性本身就依赖于词库的完整性，如果词库中根本没有新词，我们又怎么能信任分词结果呢？此时，一种大胆的想法是，首先不依赖于任何已有的词库，仅仅根据词的共同特征，将一段大规模语料中可能成词的文本片段全部提取出来，不管它是新词还是旧词。然后，再把所有抽出来的词和已有词库进行比较，不就能找出新词了吗？有了抽词算法后，我们还能以词为单位做更多有趣的数据挖掘工作。这里，我所选用的语料是人人网 2011 年 12 月前半个月部分用户的状态。非常感谢人人网提供这份极具价值的网络语料。

要想从一段文本中抽出词来，我们的第一个问题就是，怎样的文本片段才算一个词？大家想到的第一个标准或许是，看这个文本片段出现的次数是否足够多。我们可以把所有出现频数超过某个阈值的片段提取出来，作为该语料中的词汇输出。不过，光是出现频数高还不够，一个经常出现的文本片段有可能不是一个词，而是多个词构成的词组。在人人网用户状态中，“的电影”出现了 389 次，“电影院”只出现了 175 次，然而我们却更倾向于把“电影院”当作一个词，因为直觉上看，“电影”和“院”凝固得更紧一些。

为了证明“电影院”一词的内部凝固程度确实很高，我们可以计算一下，如果“电影”和“院”真的是各自独立地在文本中随机出现，它俩正好拼到一起的概率会有多小。在整个 2400 万字的数据中，“电影”一共出现了 2774 次，出现的概率约为 0.000113。“院”字则出现了 4797 次，出现的概率约为 0.0001969。如果两者之间真的毫无关系，它们恰好拼在了一起的概率就应该是 $0.000113 \times 0.0001969$ ，约为 2.223×10^{-8} 次方。但事实上，“电影院”在语料中一共出现了 175 次，出现概率约为 7.183×10^{-6} 次方，是预测值的 300 多倍。类似地，统计可得“的”字的出现概率约为 0.0166，因而“的”和“电影”随机组合到了一起的理论概率值为 0.0166×0.000113 ，约为 1.875×10^{-6} ，这与“的电影”出现的真实概率很接近——真实概率约为 1.6×10^{-5} 次方，是预测值的 8.5 倍。计算结果表明，“电影院”更可能是一个有意义的搭配，而“的电影”则更像是“的”和“电影”这两个成分偶然拼到一起的。

当然，作为一个无知识库的抽词程序，我们并不知道“电影院”是“电影”加“院”得来的，也并不知道“的电影”是“的”加上“电影”得来的。错误的切分方法会过高地估计该片段的凝固程度。如果我们把“电影院”看作是“电”加“影院”所得，由此得到的凝固程度会更高一些。因此，为了算出一个文本片段的凝固程度，我们需要枚举它的凝固方式——这个文本片段是由哪两部分组合而来的。令 $p(x)$ 为文本片段 x 在整个语料中出现的概率，那么我们定义“电影院”的凝固程度就是 $p(\text{电影院})$ 与 $p(\text{电}) \cdot p(\text{影院})$ 比值和 $p(\text{电影院})$ 与 $p(\text{电影}) \cdot p(\text{院})$ 的比值中的较小值，“的电影”的凝固程度则是 $p(\text{的电影})$ 分别除以 $p(\text{的}) \cdot p(\text{电影})$ 和 $p(\text{的电}) \cdot p(\text{影})$ 所得的商的较小值。

可以想到，凝固程度最高的文本片段就是诸如“蝙蝠”、“蜘蛛”、“彷徨”、“忐忑”、“玫瑰”之类的词了，这些词里的每一个字几乎总是会和另一个字同时出现，从不在其他场合中使用。

光看文本片段内部的凝固程度还不够，我们还需要从整体来看它在外部的表现。考虑“被子”和“辈子”这两个片段。我们可以说“买被子”、“盖被子”、“进被子”、“好被子”、“这被子”等等，在“被子”前面加各种字；但“辈子”的用法却非常固定，除了“一辈子”、“这辈子”、“上辈子”、“下辈子”，基本上“辈子”前面不能加别的字了。“辈子”这个文本片段左边可以出现的字太有限，以至于直觉上我们可能会认为，“辈子”并不单独成词，真正成词的其实是“一辈子”、“这辈子”之类的整体。可见，文本片段的自由运用程度也是判断它是否成词的重要标准。如果一个文本片段能够算作一个词的话，它应该能够灵活地出现在各种不同的环境中，具有非常丰富的左邻字集合和右邻字集合。

“信息熵”是一个非常神奇的概念，它能够反映知道一个事件的结果后平均会给你带来多大的信息量。如果某个结果的发生概率为 p ，当你知道它确实发生了，你得到的信息量就被定义为 $-\log(p)$ 。 p 越小，你得到的信息量就越大。如果一颗骰子的六个面分别是 1、1、

1、2、2、3，那么你知道了投掷的结果是1时可能并不会那么吃惊，它给你带来的信息量是 $-\log(1/2)$ ，约为0.693。知道投掷结果是2，给你带来的信息量则是 $-\log(1/3) \approx 1.0986$ 。关闭旧主题浏览模式 的信息量则有 $-\log(1/6) \approx 1.79$ 。但是，你只有1/2的机会得到0.693的信息量，只有1/3的机会得到1.0986的信息量，只有1/6的机会得到1.79的信息量，因而平均情况下你会得到 $0.693/2 + 1.0986/3 + 1.79/6 \approx 1.0114$ 的信息量。这个1.0114就是那颗骰子的信息熵。现在，假如某颗骰子有100个面，其中99个面都是1，只有一个面上写的2。知道骰子的抛掷结果是2会给你带来一个巨大无比的信息量，它等于 $-\log(1/100)$ ，约为4.605；但你只有百分之一的概率获取到这么大的信息量，其他情况下你只能得到 $-\log(99/100) \approx 0.01005$ 的信息量。平均情况下，你只能获得0.056的信息量，这就是这颗骰子的信息熵。再考虑一个最极端的情况：如果一颗骰子的六个面都是1，投掷它不会给你带来任何信息，它的信息熵为 $-\log(1) = 0$ 。什么时候信息熵会更大呢？换句话说，发生了怎样的事件之后，你最想问一下它的结果如何？直觉上看，当然就是那些结果最不确定的事件。没错，信息熵直观地反映了一个事件的结果有多么的随机。

我们用信息熵来衡量一个文本片段的左邻字集合和右邻字集合有多随机。考虑这么一句话“吃葡萄不吐葡萄皮不吃葡萄倒吐葡萄皮”，“葡萄”一词出现了四次，其中左邻字分别为{吃, 吐, 吃, 吐}，右邻字分别为{不, 皮, 倒, 皮}。根据公式，“葡萄”一词的左邻字的信息熵为 $-(1/2) \cdot \log(1/2) - (1/2) \cdot \log(1/2) \approx 0.693$ ，它的右邻字的信息熵则为 $-(1/2) \cdot \log(1/2) - (1/4) \cdot \log(1/4) - (1/4) \cdot \log(1/4) \approx 1.04$ 。可见，在这个句子中，“葡萄”一词的右邻字更加丰富一些。

在人人网用户状态中，“被子”一词一共出现了956次，“辈子”一词一共出现了2330次，两者的右邻字集合的信息熵分别为3.87404和4.11644，数值上非常接近。但“被子”的左邻字用例非常丰富：用得最多的是“晒被子”，它一共出现了162次；其次是“的被子”，出现了85次；接下来分别是“条被子”、“在被子”、“床被子”，分别出现了69次、64次和52次；当然，还有“叠被子”、“盖被子”、“加被子”、“新被子”、“掀被子”、“收被子”、“薄被子”、“踢被子”、“抢被子”等100多种不同的用法构成的长尾……所有左邻字的信息熵为3.67453。但“辈子”的左邻字就很可怜了，2330个“辈子”中有1276个是“一辈子”，有596个“这辈子”，有235个“下辈子”，有149个“上辈子”，有32个“半辈子”，有10个“八辈子”，有7个“几辈子”，有6个“哪辈子”，以及“n辈子”、“两辈子”等13种更罕见的用法。所有左邻字的信息熵仅为1.25963。因而，“辈子”能否成词，明显就有争议了。“下子”则是更典型的例子，310个“下子”的用例中有294个出自“一下子”，5个出自“两下子”，5个出自“这下子”，其余的都是只出现过一次的罕见用法。事实上，“下子”的左邻字信息熵仅为0.294421，我们不应该把它看作一个能灵活运用的词。当然，一些文本片段的左邻字没啥问题，右邻字用例却非常贫乏，例如“交响”、“后遗”、“鹅卵”等，把它们看作单独的词似乎也不太合适。我们不妨就把一个文本片段的自由运用程度定义为它的左邻字信息熵和右邻字信息熵中的较小值。

在实际运用中你会发现，文本片段的凝固程度和自由程度，两种判断标准缺一不可。只看凝固程度的话，程序会找出“巧克”、“俄罗”、“颜六色”、“柴可夫”等实际上是“半个词”的片段；只看自由程度的话，程序则会把“吃了一顿”、“看了一遍”、“睡了一晚”、“去了一趟”中的“了一”提取出来，因为它的左右邻字都太丰富了。

我们把文本中出现过的所有长度不超过d的子串都当作潜在的词（即候选词，其中d为自己设定的候选词长度上限，我设定的值为5），再为出现频数、凝固程度和自由程度各设定一个阈值，然后只需要提取出所有满足阈值要求的候选词即可。为了提高效率，我们可以把语料全文视作一整个字符串，并对该字符串的所有后缀按字典序排序。下表就是对“四是四四是十四四是十四四是十四”的所有后缀进行排序后的结果。实际上我们只需要在内存中存储这些后缀的前d+1个字，或者更好地，只储存它们在语料中的起始位置。

十
十四是十四十四是四十
十是十四是十四十四是四十
十是四十
十四是十四十四是四十
十四十四是四十
是十四是十四十四是四十
是十四十四是四十
是四十
是四十是十四是十四十四是四十
四十
四是十四是十四十四是四十
四是四十
四是十四十四是四十
四是十四是十四是十四十四是四十
四十四是四十

这样的话，相同的候选词便都集中在了一起，从头到尾扫描一遍便能算出各个候选词的频数和右邻字信息熵。将整个语料逆序后重新排列所有的后缀，再扫描一遍便能统计出每个候选词的左邻字信息熵。另外，有了频数信息后，凝固程度也都很好计算了。这样，我们

便得到了一个无需任何知识库的抽词算法，输入一段充分长的文本，这个算法能以大致 $O(n \cdot \log n)$ 的效率提取出可能的词来。

关闭旧主题浏览模式

对不同的语料进行抽词，并且按这些词的频数从高到低排序。你会发现，不同文本的用词特征是非常明显的。下面是对《西游记》上册的抽词结果：

行者、师父、三藏、八戒、大圣、菩萨、悟空、怎么、和尚、唐僧、老孙、溃骸、什么、沙僧、太宗、徒弟、袈裟、妖精、玉帝、今日、兄弟、公主、玄奘、陛下、宝贝、性命、晓得、门外、妖魔、光蕊、观音、花果山、土地、木叉、东土、变化、变做、伯钦、判官、多少、真君、齐天大圣、蟠桃、丞相、魏征、扯住、溃骸澳、抬头、揭谛、言语、猪八戒、兵器、吩咐、安排、叩头、清风、哪吒、左右、美猴王、钉钯、孩儿、女婿、金箍棒、二郎、东西、许多、奈何、人参果、收拾、近前、太保、明月、南海、水帘洞、门首、弼马温、李天王……

《资本论》全文：

商品、形式、货币、我们、过程、自己、机器、社会、部分、表现、没有、流通、需要、增加、已经、交换、关系、先令、积累、必须、英国、条件、发展、麻布、儿童、进行、提高、消费、减少、任何、手段、职能、土地、特殊、实际、完全、平均、直接、随着、简单、规律、市场、增长、上衣、决定、什么、制度、最后、支付、许多、虽然、棉纱、形态、棉花、法律、绝对、提供、扩大、独立、世纪、性质、假定、每天、包含、物质、家庭、规模、考察、剥削、经济学、甚至、延长、财富、纺纱、购买、开始、代替、便士、怎样、降低、能够、原料、等价物……

《圣经》全文：

以色列、没有、自己、一切、面前、大卫、知道、什么、犹太、祭司、摩西、看见、百姓、吩咐、埃及、听见、弟兄、告诉、基督、已经、先知、扫罗、父亲、雅各、永远、攻击、智慧、荣耀、临到、洁净、离开、怎样、平安、律法、支派、许多、门徒、打发、好像、仇敌、原文作、名叫、巴比伦、今日、首领、旷野、所罗门、约瑟、两个、燔祭、法老、衣服、脱离、二十、公义、审判、十二、亚伯拉罕、石头、聚集、按着、祷告、罪孽、约书亚、事奉、指着、城邑、进入、彼此、建造、保罗、应当、摩押、圣灵、惧怕、应许、如今、帮助、牲畜……

《时间简史》全文：

黑洞、必须、非常、任何、膨胀、科学、预言、太阳、观察、定律、运动、事件、奇点、坍缩、问题、模型、方向、区域、知道、开始、辐射、部分、牛顿、产生、夸克、无限、轨道、解释、边界、甚至、自己、类似、描述、最终、旋转、爱因斯坦、绕着、什么、效应、表明、温度、研究、收缩、吸引、按照、完全、增加、开端、基本、计算、结构、上帝、进行、已经、发展、几乎、仍然、足够、影响、初始、科学家、事件视界、第二、改变、历史、世界、包含、准确、证明、导致、需要、应该、至少、刚好、提供、通过、似乎、继续、实验、复杂、伽利略……

哦，对了，还有我最喜欢的，《人民日报》2000年4月新闻版的抽词结果：

发展、我们、经济、主席、江泽民、领导、建设、关系、教育、干部、企业、问题、主义、政治、群众、改革、政府、思想、加强、台湾、地区、北京、总统、世界、记者、代表、民族、组织、历史、访问、原则、努力、管理、今天、技术、市场、世纪、坚持、社会主义、财政、江泽民主席、增长、积极、精神、同志、双方、自己、友好、领导干部、进一步、基础、提高、必须、不断、制度、政策、解决、取得、表示、活动、支持、通过、研究、没有、学习、稳定、举行、欢迎、农村、生活、促进、科技、投资、科学、环境、领域、公司、情况、充分……

当然，我也没有忘记对人人网用户状态进行分析——人人网用户状态中最常出现的词是：

哈哈、什么、今天、怎么、现在、可以、知道、喜欢、终于、这样、觉得、因为、如果、感觉、开始、回家、考试、老师、幸福、朋友、时间、发现、东西、快乐、为什么、睡觉、生活、已经、希望、最后、各种、状态、世界、突然、手机、其实、那些、同学、孩子、尼玛、木有、然后、以后、学校、所以、青年、晚安、原来、电话、加油、果

然、学习、中国、最近、应该、需要、居然、事情、永远、特别、北京、别扭、伤不起、必须、呵呵、月亮、毕业、问题、谢谢、英语、生日快乐、工作、虽然、讨论、关闭旧主题浏览模式、今晚、继续、努力、有木有、记得……

事实上，程序从人人网的状态数据中一共抽出了大约 1200 个词，里面大多数词也确实都是标准的现代汉语词汇。不过别忘了，我们的目标是新词抽取。将所有抽出来的词与已有词库作对比，于是得到了人人网特有的词汇（同样按频数从高到低排序）：

尼玛、伤不起、给力、有木有、挂科、坑爹、神马、淡定、老爸、卧槽、牛逼、肿么、苦逼、无语、微博、六级、高数、选课、悲催、基友、蛋疼、很久、人人网、情何以堪、童鞋、哇咔咔、脑残、吐槽、猥琐、奶茶、我勒个去、刷屏、妹纸、胃疼、飘过、考研、弱爆了、太准了、搞基、忽悠、羡慕嫉妒恨、手贱、柯南、狗血、秒杀、装逼、真特么、碎觉、奥特曼、内牛满面、斗地主、腾讯、灰常、偶遇、拉拉、屌丝、九把刀、高富帅、阿内尔卡、魔兽世界、线代、三国杀、林俊杰、速速、臭美、花痴……

我还想到了更有意思的玩法。为什么不拿每一天状态里的词去和前一天的状态作对比，从而提取出这一天里特有的词呢？这样一来，我们就能从人人网的用户状态中提取出每日热点了！从手里的数据规模看，这是完全有可能的。我选了 12 个比较具有代表性的词，并列出了它们在 2011 年 12 月 13 日的用户状态中出现的频数（左列的数），以及 2011 年 12 月 14 日的用户状态中出现的频数（右列的数）：

下雪	33	92
那些年	139	146
李宇春	1	4
看见	145	695
魔兽	23	20
高数	82	83
生日快乐	235	210
今天	1416	1562
北半球	2	18
脖子	23	69
悲伤	61	33
电磁炉	0	3

大家可以从直觉上迅速判断出，哪些词可以算是 12 月 14 日的热词。比方说，“下雪”一词在 12 月 13 日只出现了 33 次，在 12 月 14 日却出现了 92 次，后者是前者的 2.8 倍，这不大可能是巧合，初步判断一定是 12 月 14 日真的有什么地方下雪了。“那些年”在 12 月 14 日的频数确实比 12 月 13 日更多，但相差并不大，我们没有理由认为它是当日的的一个热词。

一个问题摆在了我们面前：我们如何去量化一个词的“当日热度”？第一想法当然是简单地看一看每个词的当日频数和昨日频数之间的倍数关系，不过细想一下你就发现问题了：它不能解决样本过少带来的偶然性。12 月 14 日“李宇春”一词的出现频数是 12 月 13 日的 4 倍，这超过了“下雪”一词的 2.8 倍，但我们却更愿意相信“李宇春”的现象只是一个偶然。更麻烦的则是“电磁炉”一行，12 月 14 日的频数是 12 月 13 日的无穷多倍，但显然我们也不能因此就认为“电磁炉”是 12 月 14 日最热的词。

忽略所有样本过少的词？这似乎也不太好，样本少的词也有可能真的是热词。比如“北半球”一词，虽然它在两天里的频数都很少，但这个 9 倍的关系确实不容忽视。事实上，人眼很容易看出哪些词真的是 12 月 14 日的热词：除了“下雪”以外，“看见”、“北半球”和“脖子”也应该是热词。你或许坚信后三个词异峰突起的背后一定有什么原因（并且迫切地想知道这个原因究竟是什么），但却会果断地把“李宇春”和“电磁炉”这两个“异常”归结为偶然原因。你的直觉是对的——2011 年 12 月 14 日发生了极其壮观的双子座流星雨，此乃北半球三大流星雨之一。白天网友们不断转发新闻，因而“北半球”一词热了起来；晚上网友们不断发消息说“看见了”、“又看见了”，“看见”一词的出现频数猛增；最后呢，仰望天空一晚上，脖子终于出毛病了，于是回家路上一个劲儿地发“脖子难受”。

让计算机也能聪明地排除偶然因素，这是我们在数据挖掘过程中经常遇到的问题。我们经常需要对样本过少的项目进行“平滑”操作，以避免分母过小带来的奇点。这里，我采用的是一个非常容易理解的方法：一个词的样本太少，就给这个词的热度打折扣。为了便于说明，我们选出四个词为例来分析。

下表截取了前四个词，右边四列分别表示各词在 12 月 13 日出现的频数，在 12 月 14 日出现的频数，在两天里一共出现的总频数，以及后一天的频数所占的比重。第三列数字是前两列数字之和，第四列数字则是第二列数字除以第三列数字的结果。最后一列应该是一个 0

下雪	33	92	125	0.736
那些年	139	146	285	0.512
李宇春	1	4	5	0.8
看见	145	695	840	0.827
(平均)			313.75	0.719

怎么做呢？我们把每个词的得分都和全局平均分取一个加权平均！首先计算出这四个词的平均总频数，为 313.75；再计算出这四个词的平均得分，为 0.719。接下来，我们假设已经有 313.75 个人预先给每个词都打了 0.719 分，换句话说每个词都已经收到了 313.75 次评分，并且所有这 313.75 个评分都是 0.719 分。“下雪”这个词则还有额外的 125 个人评分，其中每个人都给了 0.736 分。因此，“下雪”一词的最终得分就是：

下雪	$(0.736 \times 125 + 0.719 \times 313.75) / (125 + 313.75) \approx 0.724$
----	---

类似地，其他几个词的得分依次为：

那些年	$(0.512 \times 285 + 0.719 \times 313.75) / (285 + 313.75) \approx 0.62$
李宇春	$(0.8 \times 5 + 0.719 \times 313.75) / (5 + 313.75) \approx 0.7202$
看见	$(0.827 \times 840 + 0.719 \times 313.75) / (840 + 313.75) \approx 0.798$

容易看出，此时样本越大的词，就越有能力把最终得分拉向自己本来的得分，样本太小的词，最终得分将会与全局平均分非常接近。经过这么一番调整，“下雪”一词的得分便高于了“李宇春”。实际运用中，313.75 这个数也可以由你自己来定，定得越高就表明你越在意样本过少带来的负面影响。这种与全局平均取加权平均的思想叫做 Bayesian average，从上面的若干式子里很容易看出，它实际上是最常见的平滑处理方法之一——分子分母都加上一个常数——的一种特殊形式。

利用之前的抽词程序抽取出人人网每一天内用户状态所含的词，把它们的频数都与前一天的作对比，再利用刚才的方法加以平滑，便能得出每一天的热词了。我手上的数据是人人网 2011 年 12 月上半月的数据，因此我可以得出从 12 月 2 日到 12 月 15 日的热词（选取每日前 5 名，按得分从高到低）。

- 2011-12-02: 第一场雪、北京、金隅、周末、新疆
- 2011-12-03: 荷兰、葡萄牙、死亡之组、欧洲杯、德国
- 2011-12-04: 那些年、宣传、期末、男朋友、升旗
- 2011-12-05: 教室、老师、视帝、体育课、质量
- 2011-12-06: 乔尔、星期二、摄影、经济、音乐
- 2011-12-07: 陈超、星巴克、优秀、童鞋、投票
- 2011-12-08: 曼联、曼城、欧联杯、皇马、冻死
- 2011-12-09: 保罗、月全食、交易、火箭、黄蜂
- 2011-12-10: 变身、罗伊、穿越、皇马、巴萨
- 2011-12-11: 皇马、巴萨、卡卡、梅西、下半场
- 2011-12-12: 淘宝、阿内尔卡、双十二、申花、老师
- 2011-12-13: 南京、南京大屠杀、勿忘国耻、默哀、警报
- 2011-12-14: 流星雨、许愿、愿望、情人节、几颗
- 2011-12-15: 快船、保罗、巴萨、昨晚、龙门飞甲

看来，12 月 14 日果然有流星雨发生。

注意，由于我们仅仅对比了相邻两天的状态，因而产生了个别实际上是由工作日/休息日的区别造成的“热词”，比如“教室”、“老师”、“星期二”等。把这样的词当作热词可能并不太妥。结合上周同日的数据，或者干脆直接与之前整个一周的数据来对比，或许可以部分地解决这一问题。

事实上，有了上述工具，我们可以任意比较两段不同文本中的用词特点。更有趣的是，人人网状态的大多数发布者都填写了性别和年龄的个人信息，我们为何不把状态重新分成男性和女性？关闭旧主题浏览模式且，挖掘出不同属性的人都爱说什么？要知道，在过去，这样的问题需要进行大规模语言统计调查才能回答！然而，在互联网海量用户生成内容的支持下，我们可以轻而易举地挖掘出答案来。

我真的做了这个工作（基于另一段日期内的数据）。男性爱说的词有：

兄弟、篮球、男篮、米兰、曼联、足球、蛋疼、皇马、比赛、国足、超级杯、球迷、中国、老婆、政府、航母、踢球、赛季、股市、砸蛋、牛逼、铁道部、媳妇、国际、美国、连败、魔兽、斯内德、红十字、经济、腐败、程序、郭美美、英雄、民主、鸟巢、米兰德比、官员、内涵、历史、训练、评级、金融、体育、记者、事故、程序员、媒体、投资、事件、社会、项目、伊布、主义、决赛、操蛋、纳尼、领导、喝酒、民族、新闻、言论、和谐、农民、体制、城管……

下面则是女性爱说的词：

一起玩、蛋糕、加好友、老公、呜呜、姐姐、嘻嘻、老虎、讨厌、妈妈、呜呜呜、啦啦啦、便宜、减肥、男朋友、老娘、逛街、无限、帅哥、礼物、互相、奶茶、委屈、各种、高跟鞋、指甲、城市猎人、闺蜜、巧克力、第二、爸爸、宠物、箱子、吼吼、大黄蜂、狮子、胃疼、玫瑰、包包、裙子、游戏、遇见、嘿嘿、灰常、眼睛、各位、妈咪、化妆、玫瑰花、蓝精灵、幸福、陪我玩、任务、怨念、舍不得、害怕、狗狗、眼泪、温暖、面膜、收藏、李民浩、神经、土豆、零食、痘痘、戒指、巨蟹、晒黑……

下面是 90 后用户爱用的词：

加好友、作业、各种、乖乖、蛋糕、来访、卧槽、通知书、麻将、聚会、补课、欢乐、刷屏、录取、无限、互相、速度、一起玩、啦啦啦、晚安、求陪同、基友、美女、矮油、巨蟹、五月天、第二、唱歌、老虎、扣扣、啧啧、帅哥、哈哈哈、尼玛、便宜、苦逼、斯内普、写作业、劳资、孩纸、哎哟、炎亚纶、箱子、无聊、求来访、查分、上课、果断、处女、首映、屏蔽、混蛋、暑假、吓死、新东方、组队、下学期、陪我玩、打雷、妹纸、水瓶、射手、搞基、吐槽、同学聚会、出去玩、呜呜、白羊、表白、做作业、签名、姐姐、停机、伏地魔、对象、哈哈、主页、情侣、无压力、共同、摩羯、碎觉、肿么办……

下面则是 80 后用户爱用的词：

加班、培训、周末、工作、公司、各位、值班、砸蛋、上班、任务、公务员、工资、领导、包包、办公室、校内、郭美美、时尚、企业、股市、新号码、英国、常联系、实验室、论文、忙碌、项目、部门、祈福、邀请、招聘、顺利、朋友、红十字、男朋友、媒体、产品、标准、号码、存钱、牛仔裤、曼联、政府、简单、立秋、事故、伯明翰、博士、辞职、健康、销售、深圳、奶茶、搬家、实验、投资、节日快乐、坚持、规则、考验、生活、体制、客户、发工资、忽悠、提供、教育、处理、惠存、沟通、团购、缺乏、腐败、启程、红十字会、结婚、管理、环境、暴跌、服务、变形金刚、祝福、银行……

不仅如此，不少状态还带有地理位置信息，因而我们可以站在空间的维度对信息进行观察。这个地方的人都爱说些什么？爱说这个词的人都分布在哪里？借助这些包含地理位置的签到信息，我们也能挖掘出很多有意思的结果来。例如，对北京用户的签到信息进行抽词，然后对于每一个抽出来的词，筛选出所有包含该词的签到信息并按地理坐标的位置聚类，这样我们便能找出那些地理分布最集中的词。结果非常有趣：“考试”一词集中分布在海淀众高校区，“天津”一词集中出现在北京南站，“逛街”一词则全都在西单附近扎堆。北京首都国际机场也是一个非常特别的地点，“北京”、“登机”、“终于”、“再见”等词在这里出现的密度极高。

从全国范围来看，不同区域的人也有明显的用词区别。我们可以将全国地图划分成网格，统计出所有签到信息在各个小格内出现的频数，作为标准分布；然后对于每一个抽出来的词，统计出包含该词的签到信息在各个小格内出现的频数，并与标准分布进行对比（可以采用余弦距离等公式），从而找出那些分布最反常的词。程序运行后发现，这样的词还真不少。一些明显具有南北差异的词，分布就会与整个背景相差甚远。例如，在节假日的时候，“滑雪”一词主要在北方出现，“登山”一词则主要在南方出现。地方特色也是造成词语分布差异的一大原因，例如“三里屯”一词几乎只在北京出现，“热干面”一词集中出现在武汉地区，“地铁”一词明显只有个别城市有所涉及。这种由当地人的用词特征反映出来的真实的地方特色，很可能是许多旅游爱好者梦寐以求的信息。另外，方言也会导致用词分布差异，例如“咋这

其实，不仅仅是发布时间、用户年龄、用户性别、地理位置这四个维度，我们还可以对浏览器、用户职业、用户活跃度、用户行为偏好等各种各样的维度进行分析，甚至可以综合考虑以上维度，在某个特定范围内挖掘热点事件，或者根据语言习惯去寻找出某个特定的人群。或许这听上去太过理想化，不过我坚信，有了合适的算法，这些想法终究会被一一实现。

242 条评论

- 

楼层: 沙发 | **JEB** 说:
太强大了...
- 

楼层: 板凳 | **stuart** 说:
牛逼。
- 

楼层: 地毯 | **TomLenen** 说:
迫不及待的想按这种方法分析自己的作品，看看自己有多奇葩...
- 

楼层: 地板 | **zhj** 说:
迫不及待地求一个在线文章分析器.....
- 

楼层: 地下室 | 放了假的人 说:
也许这个事情.....社交网站正在做.....毕竟社交网站汇聚了太多日常信息，为了提供更好地服务一定会不断地发展对用户所发布信息进行分析的技术，但是.....怕是到时候社交网站成为网络营销的数据库吧.....
- 

楼层: 地基 | **hdcn** 说:
这篇文章很有意思，比纯数学的主题更引人入胜啊.....
- 

楼层: 地壳 | **eaglefantasy** 说:
太强大了!!! 我上学期挑战杯的时候也潜涉了一下SNS的文本发掘。。你这个做得实在是太赞了!!!
- 

楼层: 地幔 | **mz** 说:
太强大了，回头做成js程序
- 

楼层: 地核 | **WinL** 说:
地核是我的。
- 

楼层: 10楼 | **vuryleo** 说:
讚一個~話說推薦系統是不是也可以這麼做?
- 

楼层: 11楼 | 梦里醉逍遥 说:
这篇文章太赞了!
- 

楼层: 12楼 | **sqybi** 说:
不得不说太赞了。。。
- 楼层: 12a楼 | **leokan** 说:
好久没来，一来就看到如此牛逼的文章



楼层：14楼 | **wour** 说：
学习一下。。

关闭旧主题浏览模式



楼层：15楼 | **winsty** 说：
分析热度这个事情可以试试dynamic topic model



楼层：16楼 | **仙雾** 说：
这两种方法我在计算机自然语言处理一书语句分词中都见过 可是M67大牛由此想到的抽取新词的方法将其运用的出神入化啊 膜拜



楼层：17楼 | **fmt** 说：
太强了，收获很大



楼层：18楼 | **XuLanTu** 说：
赞！！



楼层：19楼 | **Ernest** 说：
一日更兩篇文，！！



楼层：20楼 | **Ernest** 说：
看到人人网状态那部分我笑喷了。。。



楼层：21楼 | **blue** 说：
很有意义！



楼层：22楼 | **Maigo** 说：
太准了！很有信息量的一篇文章啊！



楼层：23楼 | **刘刘漂啊漂** 说：
牛逼到爆炸



楼层：24楼 | **mine260309** 说：
很好很强大！充分体现了数学在作用啊，呵呵



楼层：25楼 | **Seter** 说：
HJF也讲过类似的。。。学长明年求蹭饭。。



楼层：26楼 | **TIm Shen** 说：
Awesome..!竟然连我都看懂了...



楼层：27楼 | **BYVoid** 说：
非常好的介绍



楼层：28楼 | **一棵白菜** 说：
赞一个~~

楼层：29楼 | **少林旋龟** 说：



楼上堆满大神。

关闭旧主题浏览模式



楼层: 30楼 | **tanglei** 说:
data mining啊, 我想学的方向, 不知道接下来的读硕阶段能有时间么噢。



楼层: 31楼 | **Nemo** 说:
我想说, 我了解的数据智能真是渣渣啊!



楼层: 32楼 | **xhinker** 说:
太感谢了, 非常非常有启发



楼层: 33楼 | **Hanxiao** 说:
在西游记中的词“溃骸澳”应该是乱码统计结果吧, Google一下发现溃骸澳是一个出现频率很高的乱码字。



楼层: 34楼 | **胡搞** 说:
启发很大, 但是中文分词还是要依赖庞大的词库呀



楼层: 35楼 | **turfkids** 说:
非常好的介绍文章!



楼层: 36楼 | **小木羊** 说:
虽然我已36楼, 但看了之后还是忍不住要说赞一个



楼层: 37楼 | **Desmond** 说:
很强大很有意思。。。。



楼层: 38楼 | **pl** 说:
溃骸澳 == %E6%BA%83%E9%AA%B8%E6%BE%B3
这东西到底是什么
楼层: 33楼



楼层: 39楼 | **Sten** 说:
好劲! 获益匪浅!



楼层: 40楼 | **mercivi** 说:
很有意思! 对数据分析有点兴趣了~~~



楼层: 41楼 | **as_you_2007** 说:
这好强大啊, 太牛逼了~~~



楼层: Answer to Life, the Universe, and Everything | **[;e^{ipi}+1;]** 说:
溃骸澳
应该是什么 奴婢



楼层: 43楼 | **slash** 说:
对语料库进行排序应该是用的suffix array算法吧 看着很熟悉

楼层: 44楼 | **纠结** 说:



呃，文章很牛，非常厉害，刊登到程序员的那个也合适。确实信息很多。河蟹是因为“男性爱的词”，“女性爱的词”这样的么。

关闭旧主题浏览模式



楼层：45楼 | **Crazykoo** 说：
看了楼主的分析突然觉得挖掘数据超有趣~~



楼层：46楼 | **waveacme** 说：
原理数据挖掘这么好玩



楼层：47楼 | **waveacme** 说：
原来



楼层：48楼 | **medrifter** 说：
非常棒的工作！



楼层：49楼 | 阿弋聆风 说：
把天涯杂谈版，凯迪上的文章和评论分析一下，就知道最新的敏感词库了！



楼层：50楼 | 阿弋聆风 说：
我想到，把天涯凯迪上的文章和评论分析一下，就知道最新的敏感词库了！



楼层：51楼 | 东狗 说：
抽取新词的方法很像英文里按照mutual information来找词组的方法，这相关的也还有不少的算法可以借鉴~



楼层：52楼 | **reta** 说：
很棒的文章！



楼层：53楼 | **springbarley** 说：
其实分析的过程是亮点，楼主深厚的数学功底分析语言学规律是亮点。
新词发现用到了mutual information，其实还有很多方法。
suffix数组的算法也是常用的n-gram统计和计算概率的方法。



楼层：54楼 | 王者自由 说：
真的是非常强大有用的算法啊！



楼层：55楼 | **fountainhead** 说：
信息量的定义 $-\log(p)$ 中log的底数未标明，还是说就是 $-\lg(p)$



楼层：56楼 | **fountainhead** 说：
信息量的定义为 $-\log(p)$ ，但log的底数未标明，还是说就是 $-\lg(p)$ 。




楼层：57楼 | 嘟嘟宝贝 说：
任何东西只要钻研下去不懈努力，都会有成果。楼主加油！




楼层：58楼 | **wzyboy** 说：
很劲爆的文章，分析结果非常有意思。
分析过程也非常强大。

楼层：59楼 | **brazilla** 说：

关闭旧主题浏览模式



感觉挖掘的结论通常的认知符合
或者说，结论是平凡的




楼层：60楼 | **exp618** 说：
膜拜，神算法，神结果……




楼层：61楼 | **rihkddd** 说：
膜拜。




楼层：62楼 | **zyx199199** 说：
楼主，如果12月14的热词真是如楼主所分析的，为什么“流星雨”一词没有成为热词？理论上来说，这个词应该比北半球什么的更常出现吧




楼层：63楼 | **hhtytommy** 说：
M67神牛的这篇文章突然风靡整个人人




楼层：64楼 | **colin** 说：
非常的感兴趣，期待楼主能再些几篇对该问题深入剖析的文章




楼层：65楼 | **ymyfish** 说：
楼主你好，我最近刚刚开始接触文本挖掘的领域，跟老师做一些opinion分析的工作，对你文中提的那个nlog(n)的算法很感兴趣，但是对“十四是十四。。。”那个例子没看懂。请问这个算法有更具具体介绍的文章吗？如果有可以推荐给我看看吗？另外还有一个问题，Mutual Information和Entropy这两个阈值常设置为多少呢？有没有什么经验值，还是说取结果TOP几这样呢？




楼层：66楼 | **chengdu** 说：
标点符号怎么处理的？



楼层：67楼 | **Catastrophe** 说：
从头到尾扫描一遍便能算出各个候选词的频数和右邻字信息熵
具体怎么做？ 没想明白




楼层：68楼 | **xiayang** 说：
仔细看完，喜欢到爆。




楼层：69楼 | **Maigo** 说：
To 33,39,43楼：
“溃骸澳”是由于汉字编码的两个字节结合错误造成的乱码。
在汉字字符集GB2312中，“溃骸澳”的编码为C0 A3 BA A1 B0 C4，如果拿出中间四个字节A3 BA A1 B0，得到的内容是冒号 加上引号。这的确是一个高频字符串。
另外，“道”字的编码为B5 C0，其第二个字节与“溃”字的第一个字节相同，所以 冒号 前面极有可能是“道”字。



楼层：70楼 | **TomHall** 说：
Maigo大牛果然厉害~



楼层：71楼 | **风过无痕** 说：
赞



楼层：72楼 | **乐天谷** 说：
matrix67，您好！我是江苏教育出版社的一名报刊编辑，关注您的博客很久了，也拜读了您的新书《思考的乐趣》。在其中看到很多适合中学生阅读的文章，如果愿意，是否可以转载到我们的教辅报刊上。期待您的回复，也希望作为一个数学爱好者和您交个朋友。我的qq：330607547，email：sdxxblh@yahoo.cn.期待您的答复！



楼层：73楼 | **kn** 说：

算法簡潔而聰明, 成效很不錯... 讚一個!

关闭旧主题浏览模式



楼层：74楼 | **chengdu** 说：

不会吧，其实看看这篇论文Using Suffix Arrays to Compute Term

Frequency and Document Frequency for All Substrings in a Corpus，就知道这篇文章的出处了。要说创新只能说是贝叶斯平均了，但是imdb的评分用了很多年了。



楼层：75楼 | **66** 说：

【潰骸澳】的编码为 [C0 A3 BA A1 B0 C4]，
最有可能是 [B5 C0 A3 BA A1 B0 C4 E3] 错位造成的，
对应这四个字符【道：“你”】
比如【祖师披衣盘坐，喝道：“你这猢猻，】



楼层：76楼 | **wxn** 说：

76楼，matrix是实习又不是发paper，他又没说是方法创新，用同样的方法分析人人不也很有意思么



楼层：77楼 | **chengdu** 说：

好吧，我的意思说按照他的说法没法实现，论文讲的还是不错的。这个只是看看热闹吧，科普一下。



楼层：78楼 | **rl** 说：

太牛了！



楼层：79楼 | **staticor** 说：

这文章真的对我很有帮助。。 谢谢



楼层：80楼 | **躬行还恨赋才微** 说：

分析人民日报，我这种编程菜鸟，不苛求时间，我也能捏造个程序，把词全部抓对，抓出来！

拜网络审查所赐，我修成了谐音大法！泥马，谐音，你程序，能智能到这份上？

伟大，伟大，光荣，官荣，正确，挣券，机器如何判断意思？

如果说机器连语气都能猜出来！我服了你！

我没有攻击你的意思啊！我快归位了！我能攻击谁？

还有一高手，机器学习的！人家去美国麻省理工了！

别分析什么汉语，搞事情，挑个多快好省的！

指不定哪天，英语成了大陆，官方语言，哪你是何苦来哉呢？



楼层：81楼 | **花楹** 说：

我想起了一首诗：东风何处是人间。山人何处君不见，东风一花倚栏杆。（不知道是什么意思的童鞋的请搜索关键字“东风何处是人间”和“山人何处君不见，东风一花倚栏杆”）

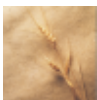


楼层：82楼 | **crown** 说：

这到让我想起之前，想做过一个地区新闻的数据库，比如爆炸，强奸，这样的新闻和地点相关起来，最初这样想是因为某一天走到中石化的大楼，想谈前两天有一个1800万吊灯的新闻就是在这个大厅里，不知道能不能看到呢。当然没看到，隔很远。

不过现在想来，如果能做地区性的这样的一个新闻统计，也可以结合当地警察部门的数据，看看某些地区是不是频发某一类事情，那么这类事情肯定在这这里有一定的根源，然后就可以解决这个问题了。

当然细想来说，当地的人很多都知道是出了什么问题，只是他们不想解决罢了。




楼层：83楼 | **coodoing** 说：

太赞了
尤其是信息熵分析“文本片段的运用自由度”

楼层：84楼 | **abc_123** 说：

特地来膜拜一个，太牛逼！



楼层：85楼 | **spk** 说：


关闭旧主题浏览模式

一个小问题 单字实词就提取不出来了
而且这几种检测方法似乎很难混淆的问题




楼层：86楼 | **Eclipse~** 说：

请教一下为什么信息量的定义是 $-\log(p)$? 谢谢！！




楼层：89楼 | **cywbsh** 说：

膜拜啊




楼层：90楼 | **hopliu** 说：

真是大牛啊 这么复杂的东西讲的这么简单 比单纯那些数据挖掘的书好了不止多少倍




楼层：91楼 | **肉牛** 说：

真是太牛逼了




楼层：92楼 | **chrysalis** 说：

“四是四十是十四四是十四四是四十”
这个例子有必要举得这么变态么 -_-#




楼层：93楼 | **fs302** 说：

向楼上的大牛们看齐~leokan、BYVoid.....好久不见！




楼层：94楼 | **美金兄弟连** 说：

相当有水准啊，比看书本强多，实例分析还是比较好理解的




楼层：95楼 | **允诺永在** 说：

M大牛真是越来越牛气了~




楼层：96楼 | **Iz** 说：

真是大牛。对你膜拜啊、、




楼层：97楼 | **jimdt69** 说：

我的研究方向是隐私保护数据挖掘，不知道M牛对这个领域有没有研究？




楼层：98楼 | **www** 说：

让我有了灵感，准备写论文给我党我政府伟大的防火墙做个敏感词监控思路研讨




楼层：99楼 | **pty** 说：

又学到了很多信息处理的知识。~




楼层：100楼 | **Rowson** 说：

赞！
M对计算动词理论的应用方面有什么好的见解？




楼层：101楼 | **怪兽小姐** 说：


赞！




楼层：102楼 | **whyhowkey** 说：
兄弟, 有没有看过 the joys of stat 的最后一段? 那! [关闭旧主题浏览模式](#) 用的是facebook




楼层：103楼 | **Mchao0** 说：
真好.




楼层：104楼 | **bobo** 说：
好厉害




楼层：105楼 | **hoszbh** 说：
“大二”、“教务”、“一生”、“作工”、“学说”、“导出”、“生成”这些词，用你的凝固程度公式计算，结果凝固程度会很低，我这边算得的结果低于1，这些类似缩略语的词，也许是无监督的方法无法解决的问题



楼层：106楼 | **hoszbh** 说：
“大二”、“教务”、“学报”、“一生”、“作工”、“学说”、“导出”、“生成”这些词，用你的凝固程度公式计算，结果凝固程度会很低，我这边算得的结果低于1，这些类似缩略语的词，也许是无监督的方法无法解决的问题




楼层：107楼 | **混混** 说：
大牛，我由衷地佩服你！




楼层：108楼 | **SilRadius** 说：
楼主厉害，这个问题我想了很久也没有思路，终于看到解决方案了。




楼层：109楼 | **三江小渡** 说：
留名。。。对来晚了表示羞愧。。




楼层：110楼 | **fero2004** 说：
牛气冲天阿



楼层：111楼 | **ArticleSea** 说：
说得通俗易懂、深入浅出！牛！




楼层：112楼 | **ppia** 说：
看完通篇，唯有一次在脑中，统计的美妙。



楼层：113楼 | **mortimer-III** 说：
原来数学还能用在语文上，看来数学还真不是一般的强大，越来越喜欢数学了。。。。



楼层：114楼 | **mortimer-III** 说：
同《数学之美》里讲的统计语言模型有异曲同工之妙。。。



楼层：115楼 | **lnx** 说：
赞一个，曾经参与过两个项目，是关于安全登录后根据用户操作电脑的习惯重新验证用户是否合法的算法，还有在网络社区中通过数据挖掘用户文章筛法识别出哪个用户是哪个用户马甲的算法，都是基于贝叶斯学习的。lz谈到的领域涉及了一些人工智能机械学习，建议可以参考一下现在的垃圾邮件过滤算法。



楼层：116楼 | **小e** 说：
在整个 2400 万字的数据中，“电影”一共出现了 2774 次，出现的概率约为 0.000113。“院”字则出现了 4797 次，出现的概率约为 0.0001969。
这个概率怎么得到的？ $2774/2400kw = 0.000115$ $4797/2400kw = 0.0001999$
词出现的概率除以字数？

关闭旧主题浏览模式

楼层：118楼 | **accry** 说：

赞，最近在学这个，真的很强大

楼层：119楼 | **CheeseZH** 说：

interesting~

楼层：120楼 | **feeldead** 说：

后半部分比较两个文本的词频差异和生物信息里面的“寻找差异表达基因”的问题太象了，只不过此处问题更简单。对于观测次数太少的变化给一定的惩罚，有一个工具GFOLD可以很好的解决这个问题。

楼层：121楼 | **yh** 说：

被人转到acfun了。。

楼层：122楼 | **cj** 说：

这篇文章太有意思了，海量数据的挖掘丫~~~有意思！

楼层：123楼 | **sea** 说：

提个简单的问题：

“在整个 2400 万字的数据中，“电影”一共出现了 2774 次，出现的概率约为 0.000113。”

概率真的是2774/24000000??? 需要考虑“电影”是两个字吗？

我想了一下，好像这样是正确的。

问题是：我用windows xp自带的计数器算的，2774/24000000，为什么总是 0.000115？左右。感觉这个还是有差异的

楼层：124楼 | **计算概率有误** 说：

在整个 2400 万字的数据中，“电影”一共出现了 2774 次，出现的概率约为 0.000113。”

如果按你的算法就是2774/2400w

假设一个特殊情况，全文只有一个词“电影”则“电影”出现的概率居然是50%。请纠正。

楼层：125楼 | **计算概率有误** 说：

“电影”一共出现了 2774 次，出现的概率约为 0.000113。“院”字则出现了 4797 次，出现的概率约为 0.0001969。如果两者之间真的毫无关系，它们恰好拼在了一起的概率就应该是 $0.000113 \times 0.0001969$ ，约为 2.223×10^{-8} 次方。

这个概率并不是拼在一起的概率，而是两个字同时出现在一篇文章中(并不一定相邻出现)的概率

楼层：126楼 | **v603** 说：

说的清晰明了，不过可以看出背后有不少工作在里面，笔者的功底挺强的，赞一个！

楼层：127楼 | **老狼** 说：

其实你所描述的抽词这个思想早就有人实现了，<http://www.doc88.com/p-671609138482.html>。你所说的“内部凝固程度”和“外部紧密程度”只不过是偷换概念而已，其实就是字的互信息和左右熵信息。

楼层：128楼 | **Acemt** 说：

不错

看着挺爽的

楼层：129楼 | **calf** 说：

文中有这么一段：“对不同的语料进行抽词，并且按这些词的频数从高到低排序。你会发现，不同文本的用词特征是非常明显的。下面是对《西游记》上册的抽词结果：……”。请问这里在统计词的频数用的是什么方法？是直接类似于字符串查找的方式查找这个词在文章中出现的次数？还是先用分词程序加词表进行分词，对分词结果进行统计？如果是前者，怎么解决切分歧义？如果是后者，怎么给分词程序提供这些词语的统计信息？

楼层：130楼 | **飘零** 说：

真是可惜没有一键分享的功能让这篇文章传播的更远，向楼主这样的Geek致敬！

楼层：131楼 | **D.S.Qiu** 说：

关闭旧主题浏览模式



好像没有看到什么新意，求解.....



楼层：132楼 | **cervelo** 说：
不得不说太赞了。。。



楼层：133楼 | **t** 说：
太强大了！天才！



楼层：134楼 | **sxw** 说：
我只能有膜拜来表达自己的感受了。ORZ，今晚受益匪浅额。



楼层：135楼 | **SiNZeRo** 说：
太赞了 您的讲座



楼层：136楼 | **Penelope** 说：
女人表示根本不认识李民浩.....



楼层：137楼 | **frank5433** 说：
这是我梦寐以求想做的事情，苦于没有数据，自己把自己好友数据收集了样本量还是不够。加油做下去！加油~~



楼层：139楼 | **pandada8** 说：
大数据时代的精确营销= =



楼层：140楼 | **ywisax** 说：
博主你好，我转载了你的文章在我的博客上<http://luzhongpeng.sinaapp.com/blog/article-9.html>，如果转载有问题的话，可以提醒我删除或者修改。很喜欢博主这个样的geek精神，哈哈。



楼层：141楼 | **3434343** 说：
其实我都按照大牛文章说的实现了一个，楼上居然还有人说不可能实现。不过效果不甚理想，而且貌似算法的复杂度达不到 $n \log n$ ，应该是 n^2



楼层：143楼 | **斑马行空** 说：
写的很好！特别是在抽词的那一部分，受益匪浅..话说贝叶斯平均的方法我也用过，今天才知道我用的原来是贝叶斯平均，是不是可以算作我自己发明的？哈哈~



楼层：144楼 | **wxuan** 说：
简单、有效，赞！



楼层：145楼 | **Yu** 说：
很赞得想法
以及,楼上惊现各种大牛



楼层：146楼 | **snowindll** 说：
此算法内存消耗很大，并且很难并行化，所以大内存的服务器是必须的。



楼层：147楼 | **余争** 说：
我把楼主的算法用python实现，并进行一些必要的优化，并加入到我的分词库里。开源的分词库地址：
<https://github.com/jannson/yaha>
实现在 `yaha/wordmaker.py` 里
实用示例在 `tests/test_cutor.py` 里
欢迎大家进行测试。6M以下的文本问题不大，如若要分析更大的文本，后续会添加一个c++实现的版本，测试发现比python快 10倍



楼层：151楼 | **quanquan127** 说：

看了你的文章，对于这段话“要想从一段文本中抽取出有意义的文本片段才算一个词？大家想到的第一个标准或许是，看这个文本片段出现的次数是否足够多。”，我想问，你是如何把句子划分成一个个文本片段的，难道是用分词器，还是用什么方式？



楼层：152楼 | **Phil** 说：

不得不留言，太有意思了，也很有意义，大数据啊！



楼层：153楼 | **Alvan** 说：

文中出现的信息熵/量的数值计算结果不对喔



楼层：154楼 | **wukun** 说：

我对凝合度的定义不太理解：
就算“蝙蝠”总是凝合在一起，但“蝙蝠”的凝合度取决于它在语料中的概率，这与直觉不太匹配。似乎不是well-defined的一个度量。至少应该满足：如果w1w2总是成对出现，w3w4总是成对出现，那么 $f(w1w2)=f(w3w4)$ 。



楼层：155楼 | **simon** 说：

很有用的技术文章，有空闲下时间来研究一下。。



楼层：156楼 | **zqq** 说：

注意这里log是以对数e为底，而不是10为底



楼层：157楼 | **fjzzq2002** 说：

感谢Matrix67大神，请问一下我按你说的做了，阈值大概要怎么取比较合适？



楼层：158楼 | **fjzzq2002** 说：

使我 凝固程度13.4747 自由程度4.0817
了一 凝固程度5.3014 自由程度3.0870
Matrix67大牛，为什么感觉“了一”的自由程度没有“使我”高？



楼层：160楼 | **fjzzq2002** 说：

西游记语料热词：行者、八戒、师父、三藏、一个、大圣、唐僧、沙僧、和尚、菩萨、那里、怎么、不知、我们、长老、妖精、老孙、两个、甚么、悟空、只见、国王、那怪、徒弟、呆子、闻言、与他、大王、原来、不敢、如何、不曾、小妖、正是、一声、等我、真个、这里、这等、兄弟、宝贝、今日、听得、上前、在此、取经、吃了、如今、这般、怎的、起来、孙行者、铁棒、西天、认得、天王、不能、妖怪、龙王、师徒、进去、太子、果然、东土、有些、性命、孙大圣、老爷、那呆子、观看、神通、公主、妖魔、人家、玉帝、变作、弟子、乃是、教他、猴王、哥哥、土地、道士、师兄、门外、一齐、欢喜、妖王、在那里、贫僧、云头、行李、太宗、陛下、有甚、怎生、那妖精、闻得、一座、爷爷、想是、圣僧、慌得、多少、空中、做个、叫做、变做、众僧、兵器、模样、二十、些儿、几个、不题、袈裟、心中、如此、里面、十分、一条、老魔、变化、你怎么、多时、原来是、却又、看看、怪物、到此、只听、魔王、他两个、手段、近前、一般、大仙、收了、娘娘、往西、唬得、衣服、三十、猪八戒、女子、被他、西方、齐天、吩咐、打死、师徒们、更不、仔细、孙悟空、左右、听见、金箍棒、叫声、晓得、只听得、言语、奈何、天宫、四众、真个是、观音、看时、安排、叩头、那国王……
好像和大牛的Matrix67有点不一样……



楼层：165楼 | **jason_wang** 说：

怒赞！！



楼层：166楼 | **小强** 说：

感谢！已经在Hadoop上实现



楼层：170楼 | **Anika** 说：

太赞了，大早上看到，受益

楼层：172楼 | **sing1ee** 说：



我运行的西游记的结果：

关闭旧主题浏览模式

八戒
师父
大圣
唐僧
菩萨
妖精
徒弟
兄弟
宝贝
今日
取经
如今
认得
东土
孙大圣
变作
玉帝
土地
欢喜
贫僧
变做
兵器
袈裟
十分
变化
怪物
近前
猪八戒
西方
吩咐
打死
孙悟空
左右
金箍棒
言语
观音
安排
关文
叩头
拜佛
妖邪
战兢兢
许多
抬头
扯住
筋斗
收拾
本事
皇帝
佛祖
玄奘
多官
罗刹
唤做
揭谛
御弟
磕头
花果山
文武
合掌
耳朵
哪吒
法力
保护
悟能
传旨

关闭旧主题浏览模式

母亲
悟净
昨日
群妖
出家人
躬身
包袱
南海
惊动
金刚
祥云
祥光
木叉
且休
保唐僧
光蕊
吆喝
方丈
宝杖
山坡
蟠桃
呵呵
皈依
轻轻
二郎
慈悲
水帘洞
判官
金睛
乾坤
狂风
相貌
观音菩萨
罗汉
星官
火焰
先锋
慌忙
抽身
挑着
清风
记得
尽皆
肚里
元帅
连忙
圈子
妈妈
抖擞
施主
即忙
名唤
呼呼
找寻
皇后
花园
宝剑
急急
计较
官员
层门
明月
欣然
冲撞
国丈
声响
渐渐

关闭旧主题浏览模式

玉皇
白玉
看守
纷纷
英雄
低头
供养
君臣
将军
早已
昼夜
滚滚
等候
芭蕉扇
五百年前
天竺国
天蓬
挡住
晦气
欠身
荡荡
败阵
人参果
净瓶
奶奶
心猿
使铁棒
瑞气
侍立
努力
松林
焚香
猪羊
百余
盘缠
腾云
莲花
虎豹
丈夫
世间
仙童
奉旨
极乐
架住
凌霄殿
禅堂
两班
十余
层层
强盗
惠岸
河边
石崖
筵宴
自己
莲台
几句
思量
油锅
灯火
祭赛
若肯
伽蓝
十三年
夜叉
帅众
拦住

关闭旧主题浏览模式

整衣
流沙河
火焰山
爬起来
刘洪
施威
杨柳
枪刀
欺心
罗刹女
逍遥
依旧
无穷
烟火
相逢
绣球
荆棘
南赡部洲
护持
摆列
擂鼓
暗暗
玉兔
福地
紧紧
二十八宿
五庄观
哪吒太子
太尉
忙忙
护法
捉弄
推倒
普陀
朱紫国
水晶宫
火烧
管待
翻波
耍耍
解脱
举钯
卷帘
呼唤
太平
尸首
报怨
文武多官
本寺
王父子
琉璃
男女
皇宫
绳索
金蝉
伸手
俯伏
几番
剥皮
太阳
山凹里
山神土地
巍巍
幽冥
心肝
犀牛
瑞霭

关闭旧主题浏览模式

瑶池
群猴
虎狼
走兽
身躯
重重
传令
保护唐僧
光禄寺
八百里
劈脸
发狠
善财
埋怨
宝塔
山坡下
思凡
指定
明珠
满堂
烟霞
狮驼
狼虫
瑶草
艳艳
令郎
孔雀
宇宙
御花园
掀翻
旌旗
朦胧
木母
殷勤
涧边
灼灼
童男
袖中取出
越发
东胜神洲
凤仙郡
吉凶
向南
奇花
帖儿
当驾官
拽步
沐浴
泾河
端坐
缥缈
舍利
荒山
萧瑀
蒙菩萨
蓬莱
连夜
通明殿
后园
商量
嘻嘻
宝阁
寇家
形容
扯扯
搭救
星辰

关闭旧主题浏览模式

条绳
果品
瓜果
登宝
睁睛
笑吟吟
脑后
舅爷
西域
进退
通天河
金铃
举步
令牌
刀砍
劈面
孤拐
宝象国
幢幡
彩凤
彩雾
打劫
拱手
指示
搅乱
携手
梆铃
沉吟
混沌
猿猴
芭蕉洞
落伽山
衙门
诸佛
赤脚
踏云
金光寺
参拜
后妃
四健将
囟囟
太白金星
拈香
水伯
流泪
烘烘
狐狸
獠牙
甘露
百姓
皈正
眼泪
睁眼
簇簇
能彀
虔心
金圣宫
东土唐朝
乔松
使一条
使钉耙
侮着
剑戟
北海
参禅
奉玉帝
崔判官

关闭旧主题浏览模式

异香
微微
惊醒
抵住
招呼
拯救
搀住
整治
棒架
泰山
漠漠
石屏
簇拥
紧随
菩提
野花
二将军
侍婢
修持
兜率宫
卖弄
吊桶
声振
大小官员
奔波
御妹
打探
找路
抱住
文殊
松柏
水族
江流
渺渺
珍楼
白鹤
祖宗
神祇
莲花洞
贞观
跌跌
跟随
辞别
配合
万寿山
仙翁
入里边
剿除
包裹
午时
哮吼
夜静
太宗皇帝
婴儿
寂灭
带领
彩霞
怒发
扫荡
揪住
杨柳枝
松树
毒魔
清华
潺潺
盼望
稳坐

关闭旧主题浏览模式

西牛贺洲
金山寺
万载
举目
二层门
伤害
八大金刚
养性
分身法
千余
名唤做
四大天师
嫦娥
山脚下
悠悠
慈云寺
把身子
招安
拜辞
排列
映日
柳树
梅花
波浪
消灾
清平
猱狮
玄英洞
白玉阶前
白鹿
祖居
肚皮
舍利子
苍狼
衣袖
远涉
金头揭谛
金晴山
三岛
仙卿
供献
匣儿
半盏
古树
叮咛
坠落
奉献
孤魂
宝林寺
宣至
得胜回
拄着
星宿
松篁
滔滔
灵台
炼成
烈火
玉液
王母娘娘
疑惑
稀奇
程途
红漆
至花果山
西梁国
赴蟠桃

关闭旧主题浏览模式

金线
丝绦
丹墀
举钉钯
刷洗
厨房
呻吟
四肢
围困
大唐皇帝
大慈大悲
天罡
宣召
寂寞
尊府
恩爱
慧眼
昔日
滴滴
潇潇
烟波
玉英
玉面公主
白昼
短棍
石块
石桥
祭赛国
竹节山
结彩
缓步
缭绕
翡翠
胁下
芍药
芙蓉
茶鍾
荤素
葡萄
衣钵
观音院
试试
金盃
五岳
交付
仙吏
修炼
光耀
冒犯
军士
凤凰
勉强
叙话
后宰门
太医院
如意钩
妖娆
姹女
宰相
屏风
崔巍
巅峰
巢穴
幌幌
御马监
念诵
急入里

扫净
控背躬身
救拔
日暖
旷野
查点
柴扉
江湖
波涛
瀛洲
烧纸
照验
狼虎
白米
白雪
祇园
竭力
篮儿
纳锦
绣带
翠屏
翠竹
舍死忘生
茅屋
蛾眉
血书
读书
趋步
踊跃
轮流
进贡
金丹砂
丛丛
交锋
元宵
凄凉
到跟前
剥皮亭
发慈悲
四木禽星
妙岩宫
展翅
归顺
忍疼
忒没
房屋
挨排
提兵
操演
整理
昆仑山
智渊寺
棺材
沉思
泄漏
海棠
牢拴
狮奴
白鼋
皮肉
笼蒸
芝兰
苦争
苦劝
虎口洞
二从者
交欢

关闭旧主题浏览模式

关闭旧主题浏览模式

令爱
停立
剪除
千载
南极
四季
急整衣
扫除
挑包
改作
散碎
智慧
枯藤
永寿
注定
海岳
炼丹
环眼
突兀
茅舍
虎先锋
蝼蚁
试演
赤脚大仙
连环
金圣娘娘
丝篷
光明宫
冰肌
容娇
幽禽
房檐
拄杖
敛衣
梦魂
爆燥
禽兽
追至
乘鸾
九瓣
九秋
众强人
侍卫
催趲
入后边
典膳官
压龙山
周天之
唐朝驾下
垂珠
夜游神
大厦
娇娇
嫩柳
尧舜
岐嶒
廊房
忠良
总兵官
手托着
捉得妖
捎书
捕贼
推推
改换
文华殿
斗柄

显威
曾记得
替祭
最能
杂树
枪尖
检点
涤垢
添香
温养
灵山拜佛
点查
热闹
独显
相联
箫鼓
细犬
绵花
自忖
蒙盛
藤缠
虾婆
蛇虫
赤壁
超升
邻叟
野菜

关闭旧主题浏览模式



楼层：175楼 | **myccc** 说：
这个算法思路相当赞，不过通过该算法也会增加很多我们不需要的词汇，譬如“是吗”“是吧”“但是不如”这次词，他们的PMI和信息熵也够大，不知道各位有什么好的办法剔除这些词。



楼层：177楼 | **Mr_Anderson** 说：
为什么感觉2字词，3字词和4字词的过滤阈值好像不统一？



楼层：178楼 | **QQ** 说：
楼主我是一个初入NLP领域的新手，对你的处理过程有些疑问？能否交流下



楼层：179楼 | **thinkercui** 说：
该方法存在一个潜在的难点，就是阈值设定问题。请问，楼主是如何确定阈值大小的？难道是无脑尝试？



楼层：180楼 | **陈佳** 说：
以前看过，今天再次来看看，学习一下分词。



楼层：181楼 | **xzer** 说：
你们有对日文做过测试吗？日语有跟中文同样的分词难题，你们的算法应该是不依赖于语法的（当然也许可能需要根据不同语言的特点做一些微调），所以我觉得这个算法应该很容易扩展到中日韩三国语言上来。



楼层：182楼 | **逍遥游** 说：
很强大，很浅显易懂



楼层：183楼 | **jell** 说：
强大，学习ing



楼层：184楼 | **Jason** 说：
牛逼啊~~~

- 

楼层：185楼 | seagg 说：

膜拜啊，根据楼主的思想，我用Go实现了一版，[关闭旧主题浏览模式](#) 聚度的阈值不好确定。。。
- 

楼层：188楼 | xdawn 说：

真的好好玩的样子，让人想到了无数的可能性，回去就想写一下~
- 

楼层：189楼 | pyxzzfly 说：

这种无词典的汉字提取、切分的准确率和召回率有人分析过吗？
- 

楼层：190楼 | nd 说：

膜拜&感谢。之前在数学之美里看到过分词、tf-idf、还有信息熵，67大牛给了新的体会
- 

楼层：191楼 | zc 说：

https://github.com/sing1ee/dict_build 这个简单用java实现了，看效果还不错。挺好玩的。
- 

楼层：193楼 | wencenty 说：

我想问一下，这个阈值是怎么设定的，有一定的标准吗？
- 

楼层：195楼 | 某某海 说：

想问问，“四是四十是十四四是十四四是四十”，这句话分出来的潜在的词怎么这么少？我按照d等于5的长度，分出来了48种不重复的潜的词。o(ノ □ ヽ)o
- 

楼层：196楼 | 大华 说：

不错的新词发现的方法，堪称教科书式
不知道，如果把新词扩展成本片段是否仍然有效？太短的词表意不完全，比如“看见”就没有太大表意能力，但“看见流星雨”就有相当的表意能力了
- 

楼层：197楼 | xyy 说：

语料库太大的话感觉时空代价很大啊，需要一些办法降低运算量或者做并行~
- 

楼层：198楼 | 小纸条 说：

C++11/14实现了一个，写得渣勿喷。。 <https://github.com/zouyxdut/new-words-discoverer>
- 

楼层：200楼 | zhou嗯嗯 说：

求一实现的代码spark+java的
- 

楼层：202楼 | Melody 说：

这篇文章写的并不清楚啊，比如内部凝固度时提到的概率怎么算的？词频/词个数？但是总的词数怎么算呢？分词都不确定
- 

楼层：204楼 | wanna 说：

https://github.com/sing1ee/dict_build 这个的bat文件运行就只有rawpath。请问各位大大，怎么才能运行出来呀？？
- 

楼层：205楼 | Tonywang 说：

如果一颗骰子的六个面分别是 1、1、1、2、2、3，那么你知道了投掷的结果是 1 时可能并不会那么吃惊，它给你带来的信息量是 $-\log(1/2)$ ，约为 0.693。知道投掷结果是 2，给你带来信息量则是 $-\log(1/3) \approx 1.0986$ 。知道投掷结果是 3，给你带来信息量则有 $-\log(1/6) \approx 1.79$ 。
=====
这里的信息量计算为何用的是自然数为底数而不是 2 呢？
- 楼层：206楼 | Han 说：



python 3 實現：

<https://github.com/yanghanxy/New-Word-Detector> 关闭旧主题浏览模式



楼层：207楼 | **BigNewbie** 说：

赞赞赞



楼层：208楼 | **Jtyoui** 说：

Python 3 实现了一个，写得渣勿喷：<https://gitee.com/tyoui/snsg>



楼层：209楼 | 闭眼唱歌 说：

作者有没有将这个idea发表论文呢？想引用这个想法，请问是直接引用这个网址吗？



楼层：210楼 | 菜鸡 说：

思路和隐马尔可夫很像，和统计学的最大似然有什么区别呢



楼层：211楼 | 初学者 说：

请问这个方法适合用于微博文本的新词发现吗



楼层：212楼 | **123** 说：

7年了，依然具有启发意义，神文啊。



楼层：213楼 | **05mx.com**低价网游私服**t8b8** 说：

天龙sf服务端奇迹Musf服务端魔兽sf服务端魔域sf服务端墨香sf服务端 <http://www.a3sf.com> 端天堂2sf服务端传奇3sf服务端
英雄王座sf服务端千年sf服务端征途sf服务端新魔界sf服务端骑士sf服务端 <http://www.a3sf.com> 烈焰sf服务端破天sf服务端
决战sf服务端完美世界sf服务端乱勇OLsf服务端倚天2sf服务端完美世界sf服务端征服sf服务端天堂sf服务端传世sf服务端
真封神sf服务端劲舞团sf服务端天上碑sf服务端永恒之 <http://www.a3sf.com/taocao> 塔sf服务端仙境ROsf服务端诛仙sf服
务端神泣sf服务端石器sf服务端冒险岛sf服务端惊天动地sf服务端热血江湖sf服务端问道sf服务端密传sf服务端火线任务
(Heat Project)sf服务端飞飞OLsf



楼层：214楼 | **qinliuzhuang** 说：

很牛的文章，学习了