# 谷歌吴军：数学之美

## 作者：吴军

内容简介

也许大家不相信，数学是解决信息检索和自然语言处理的最好工具。它能非常清晰地描述这些领域的实际问题并且给出漂亮的解决办法。每当人们应用数学工具解决一个语言问题时，总会感叹数学之美。我们希望利用 Google 中文黑板报这块园地，介绍一些数学工具，以及我们是如何利用这些工具来开发 Google 产品的。

也许大家不相信，数学是解决信息检索和自然语言处理的最好工具。它能非常清晰地描述这些领域的实际问题并且给出漂亮的解决办法。每当人们应用数学工具解决一个语言问题时，总会感叹数学之美。我们希望利用 Google 中文黑板报这块园地，介绍一些数学工具，以及我们是如何利用这些工具来开发 Google 产品的。

系列一： 统计语言模型 (Statistical Language Models)

Google 的使命是整合全球的信息，所以我们一直致力于研究如何让机器对信息、语言做最好的理解和处理。长期以来，人类一直梦想着能让机器代替人来翻译语言、识别语 音、认识文字（不论是印刷体或手写体）和进行海量文献的自动检索，这就需要让机器理解语言。但是人类的语言可以说是信息里最复杂最动态的一部分。为了解决 这个问题，人们容易想到的办法就是让机器模拟人类进行学习 - 学习人类的语法、分析语句等等。尤其是在乔姆斯基（Noam Chomsky 有史以来最伟大的语言学家）提出 “形式语言” 以后，人们更坚定了利用语法规则的办法进行文字处理的信念。遗憾的是，几十年过去了，在计算机处理语言领域，基于这个语法规则的方法几乎毫无突破。

其实早在几十年前，数学家兼信息论的祖师爷 香农 (Claude Shannon)就提出了用数学的办法处理自然语言的想法。遗憾的是当时的计算机条件根本无法满足大量信息处理的需要，所以他这个想法当时并没有被人们重视。七十年代初，有了大规模集成电路的快速计算机后，香农的梦想才得以实现。

首先成功利用数学方法解决自然语言处理问题的是语音和语言处理大师贾里尼克 (Fred Jelinek)。当时贾里尼克在 IBM 公司做学术休假 (Sabbatical Leave)，领导了一批杰出的科学家利用大型计算机来处理人类语言问题。统计语言模型就是在那个时候提出的。

给大家举个例子：在很多涉及到自然语言处理的领域，如机器翻译、语音识别、印刷体或手写体识别、拼写纠错、汉字输入和文献查询中，我们都需要知道一个文字序列是否能构成一个大家能理解的句子，显示给使用者。对这个问题，我们可以用一个简单的统计模型来解决这个问题。

如 果 S 表示一连串特定顺序排列的词 w1， w2，…， wn，换句话说，S 可以表示某一个由一连串特定顺序排练的词而组成的一个有意义的句子。现在，机器对语言的识别从某种角度来说，就是想知道S在文本中出现的可能性，也就是数 学上所说的S 的概率用 P(S) 来表示。利用条件概率的公式，S 这个序列出现的概率等于每一个词出现的概率相乘，于是P(S) 可展开为：

P(S) = P(w1)P(w2|w1)P(w3| w1 w2)…P(wn|w1 w2…wn-1)

其 中 P (w1) 表示第一个词w1 出现的概率；P (w2|w1) 是在已知第一个词的前提下，第二个词出现的概率；以次类推。不难看出，到了词wn，它的出现概率取决于它前面所有词。从计算上来看，各种可能性太多，无法 实现。因此我们假定任意一个词wi的出现概率只同它前面的词 wi-1 有关(即马尔可夫假设），于是问题就变得很简单了。现在，S 出现的概率就变为：

P(S) = P(w1)P(w2|w1)P(w3|w2)…P(wi|wi-1)…
(当然，也可以假设一个词又前面N-1个词决定，模型稍微复杂些。）

接 下来的问题就是如何估计 P (wi|wi-1)。现在有了大量机读文本后，这个问题变得很简单，只要数一数这对词（wi-1,wi) 在统计的文本中出现了多少次，以及 wi-1 本身在同样的文本中前后相邻出现了多少次，然后用两个数一除就可以了,P(wi|wi-1) = P(wi-1,wi)/ P (wi-1)。

也许很多人不相信用这么简单的数学模型能解决复杂的语音识别、机器翻译等问题。其实不光是常人，就连很多语言学家都曾质疑过这种方法的有效性，但事实证明，统计语言模型比任何已知的借助某种规则的解决方法都有效。比如在 Google 的中英文自动翻译中，用的最重要的就是这个统计语言模型。去年美国标准局(NIST) 对所有的机器翻译系统进行了评测，Google 的系统是不仅是全世界最好的，而且高出所有基于规则的系统很多。

现 在，读者也许已经能感受到数学的美妙之处了，它把一些复杂的问题变得如此的简单。当然，真正

实现一个好的统计语言模型还有许多细节问题需要解决。贾里尼克 和他的同事的贡献在于提出了统计语言模型，而且很漂亮地解决了所有的细节问题。十几年后，李开复用统计语言模型把 997 词语音识别的问题简化成了一个 20 词的识别问题，实现了有史以来第一次大词汇量非特定人连续语音的识别。

我是一名科学研究人员 ，我在工作中经常惊叹于数学语言应用于解决实际问题上时的神奇。我也希望把这种神奇讲解给大家听。当然，归根结底，不管什莫样的科学方法、无论多莫奇妙的解决手段都是为人服务的。我希望 Google 多努力一分，用户就多一分搜索的喜悦。

谈谈中文分词
----- 统计语言模型在中文处理中的一个应用

上回我们谈到利用统计语言模型进行语言处理，由于模型是建立在词的基础上的，对于中日韩等语言，首先需要进行分词。例如把句子“中国航天官员应邀到美国与太空总署官员开会。”

分成一串词：
中国 / 航天 / 官员 / 应邀 / 到 / 美国 / 与 / 太空 / 总署 / 官员 / 开会。

最容易想到的，也是最简单的分词办法就是查字典。这种方法最早是由北京航天航空大学的梁南元教授提出的。

用“查字典”法，其实就是我们把一个句子从左向右扫描一遍，遇到字典里有的词就标识出来，遇到复合词（比如“上海大学”）就找最长的词匹配，遇到不认识的字串就分割成单字词，于是简单的分词就完成了。这种简单的分词方法完全能处理上面例子中的句子。八十年代，哈工大的王晓龙博士把 它理论化，发展成最少词数的分词理论，即一句话应该分成数量最少的词串。这种方法一个明显的不足是当遇到有二义性 （有双重理解意思）的分割时就无能为力了。比如，对短语 “发展中国家” 正确的分割是“发展-中-国家”，而从左向右查字典的办法会将它分割成“发展-中国-家”，显然是错了。另外，并非所有的最长匹配都一定是正确的。比如“ 上海大学城书店”的正确分词应该是 “上海-大学城-书店，” 而不是 “上海大学-城-书店”。

九十年代以前，海内外不少学者试图用一些文法规则来解决分词的二义性问题，都不是很成功。90年前后，清华大学的郭进博士用统计语言模型成功解决分词二义性问题，将汉语分词的错误率降低了一个数量级。

利用统计语言模型分词的方法，可以用几个数学公式简单概括如下：
我们假定一个句子S可以有几种分词方法，为了简单起见我们假定有以下三种：
A1, A2, A3, ..., Ak,
B1, B2, B3, ..., Bm
C1, C2, C3, ..., Cn

其中，A1, A2, B1, B2, C1, C2 等等都是汉语的词。那么最好的一种分词方法应该保证分完词后这个句子出现的概率最大。也就是说如果 A1,A2,..., Ak 是最好的分法，那么 （P 表示概率）：
P (A1, A2, A3, ..., Ak） 〉 P (B1, B2, B3, ..., Bm), 并且
P (A1, A2, A3, ..., Ak） 〉 P(C1, C2, C3, ..., Cn)
因此，只要我们利用上回提到的统计语言模型计算出每种分词后句子出现的概率，并找出其中概率最大的，我们就能够找到最好的分词方法。

当然，这里面有一个实现的技巧。如果我们穷举所有可能的分词方法并计算出每种可能性下句子的概率，那么计算量是相当大的。因此，我们可以把它看成是一个动态规划（Dynamic Programming) 的问题，并利用 “维特比”（Viterbi） 算法快速地找到最佳分词。

在清华大学的郭进博士以后，海内外不少学者利用统计的方法，进一步完善中文分词。其中值得一提的是清华大学孙茂松教授和香港科技大学吴德凯教授的工作。

需 要指出的是，语言学家对词语的定义不完全相同。比如说 “北京大学”，有人认为是一个词，而有人认为该分成两个词。一个折中的解决办法是在分词的同时，找到复合词的嵌套结构。在上面的例子中，如果一句话包含“ 北京大学”四个字，那么先把它当成一个四字词，然后再进一步找出细分词 “北京” 和

“大学”。这种方法是最早是郭进在 “Computational Linguistics” （《计算机语言学》）杂志上发表的，以后不少系统采用这种方法。

一般来讲，根 据不同应用，汉语分词的颗粒度大小应该不同。比如，在机器翻译中，颗粒度应该大一些，“北京大学”就不能被分成两个词。而在语音识别中，“北京大学”一般 是被分成两个词。因此，不同的应用，应该有不同的分词系统。Google 的葛显平博士和朱安博士，专门为搜索设计和实现了自己的分词系统。

也 许你想不到，中文分词的方法也被应用到英语处理，主要是手写体识别中。因为在识别手写体时，单词之间的空格就不很清楚了。中文分词方法可以帮助判别英语单 词的边界。其实，语言处理的许多数学方法通用的和具体的语言无关。在 Google 内，我们在设计语言处理的算法时，都会考虑它是否能很容易地适用于各种自然语言。这样，我们才能有效地支持上百种语言的搜索。

对中文分词有兴趣的读者，可以阅读以下文献：

1. 梁南元
书面汉语自动分词系统
http://www.touchwrite.com/demo/LiangNanyuan-JCIP-1987.pdf

2. 郭进
统计语言模型和汉语音字转换的一些新结果
http://www.touchwrite.com/demo/GuoJin-JCIP-1993.pdf

3. 郭进
Critical Tokenization and its Properties
http://acl.ldc.upenn.edu/J/J97/J97-4004.pdf

4. 孙茂松
Chinese word segmentation without using lexicon and hand-crafted training data
http://portal.acm.org/citation.cfm?coll=GUIDE&dl=GUIDE&id=980775

前言：隐含马尔可夫模型是一个数学模型，到目前为之，它一直被认为是实现快速精确的语音识别系统的最成功的方法。复杂的语音识别问题通过隐含马尔可夫模型能非常简单地被表述、解决，让我不由由衷地感叹数学模型之妙。

自然语言是人类交流信息的工具。很多自然语言处理问题都可以等同于通信系统中的解码问题 -- 一个人根据接收到的信息，去猜测发话人要表达的意思。这其实就象通信中，我们根据接收端收到的信号去分析、理解、还原发送端传送过来的信息。以下该图就表示了一个典型的通信系统：



其中 s1，s2，s3...表示信息源发出的信号。o1, o2, o3 ... 是接受器接收到的信号。通信中的解码就是根据接收到的信号 o1, o2, o3 ...还原出发送的信号 s1，s2，s3...。

其实我们平时在说话时，脑子就是一个信息源。我们的喉咙（声带），空气，就是如电线和光缆般的信道。听众耳朵的就是接收端，而听到的声音就是传送过来的信号。根据声学信号来推测说话者的意思，就是语音识别。这样说来，如果接收端是一台计算机而不是人的话，那么计算机要做的就是语音的自动识别。同样，在计算机中，如果我们要根据接收到的英语信息，推测说话者的汉语意思，就是机器翻译；如果我们要根据带有拼写错误的语句推测说话者想表达的正确意思，那就是自动纠错。

那么怎么根据接收到的信息来推测说话者想表达的意思呢？我们可以利用叫做"隐含马尔可夫模型"（Hidden Markov Model）来解决这些问题。以语音识别为例，当我们观测到语音信号 o1,o2,o3 时，我们要根据这组信号推测出发送的句子 s1,s2,s3。显然，我们应该在所有可能的句子中找最有可能性的一个。用数学语言来描述，就是在已知 o1,o2,o3,...的情况下，求使得条件概率
P (s1,s2,s3,...|o1,o2,o3....) 达到最大值的那个句子 s1,s2,s3,...

当然，上面的概率不容易直接求出，于是我们可以间接地计算它。利用贝叶斯公式并且省掉一个常数项，可以把上述公式等价变换成

P(o1,o2,o3,...|s1,s2,s3....) * P(s1,s2,s3,...)
其中
P(o1,o2,o3,...|s1,s2,s3....) 表示某句话 s1,s2,s3...被读成 o1,o2,o3,...的可能性, 而
P(s1,s2,s3,...) 表示字串 s1,s2,s3,...本身能够成为一个合乎情理的句子的可能性，所以这个公式的意义是用发送信号为 s1,s2,s3...这个数列的可能性乘以 s1,s2,s3...本身可以一个句子的可能性，得出概率。

（读者读到这里也许会问，你现在是不是把问题变得更复杂了，因为公式越写越长了。别着急，我们现在就来简化这个问题。）我们在这里做两个假设：

第一，s1,s2,s3,... 是一个马尔可夫链，也就是说，si 只由 si-1 决定 (详见系列一)；
第二， 第 i 时刻的接收信号 oi 只由发送信号 si 决定（又称为独立输出假设, 即 P(o1,o2,o3,...|s1,s2,s3....) = P(o1|s1) * P(o2|s2)*P(o3|s3)...。
那么我们就可以很容易利用算法 Viterbi 找出上面式子的最大值，进而找出要识别的句子 s1,s2,s3,...。

满足上述两个假设的模型就叫隐含马尔可夫模型。我们之所以用"隐含"这个词，是因为状态 s1,s2,s3,...是无法直接观测到的。

隐含马尔可夫模型的应用远不只在语音识别中。在上面的公式中，如果我们把 s1,s2,s3,...当成中文，把 o1,o2,o3,...当成对应的英文，那么我们就能利用这个模型解决机器翻译问题； 如果我们把 o1,o2,o3,...当成扫描文字得到的图像特征，就能利用这个模型解决印刷体和手写体的识别。

P (o1,o2,o3,...|s1,s2,s3....) 根据应用的不同而又不同的名称，在语音识别中它被称为"声学模型" (Acoustic Model)， 在机器翻译中是"翻译模型" (Translation Model) 而在拼写校正中是"纠错模型" (Correction Model)。而P (s1,s2,s3,...) 就是我们在系列一中提到的语言模型。

在利用隐含马尔可夫模型解决语言处理问题前，先要进行模型的训练。 常用的训练方法由伯姆

（Baum）在60年代提出的，并以他的名字命名。隐含马尔可夫模型在处理语言问题早期的成功应用是语音识别。七十年代，当时 IBM 的 Fred Jelinek (贾里尼克) 和卡内基·梅隆大学的 Jim and Janet Baker (贝克夫妇，李开复的师兄师姐) 分别独立地提出用隐含马尔可夫模型来识别语音，语音识别的错误率相比人工智能和模式匹配等方法降低了三倍 (从 30% 到 10%)。 八十年代李开复博士坚持采用隐含马尔可夫模型的框架， 成功地开发了世界上第一个大词汇量连续语音识别系统 Sphinx。

我最早接触到隐含马尔可夫模型是几乎二十年前的事。那时在《随机过程》（清华"著名"的一门课）里学到这个模型，但当时实在想不出它有什么实际用途。几年后，我在清华跟随王作英教授学习、研究语音识别时，他给了我几十篇文献。我印象最深的就是贾里尼克和李开复的文章，它们的核心思想就是隐含马尔可夫模型。复杂的语音识别问题居然能如此简单地被表述、解决，我由衷地感叹数学模型之妙。

前言: Google 一直以 "整合全球信息，让人人能获取，使人人能受益" 为使命。那么究竟每一条信息应该怎样度量呢？

信息是个很抽象的概念。我们常常说信息很多，或者信息较少，但却很难说清楚信息到底有多少。比如一本五十万字的中文书到底有多少信息量。直到 1948 年，香农提出了"信息熵"(shāng) 的概念，才解决了对信息的量化度量问题。

一条信息的信息量大小和它的不确定性有直接的关系。比如说，我们要搞清楚一件非常非常不确定的事，或是我们一无所知的事情，就需要了解大量的信息。相反，如果我们对某件事已经有了较多的了解，我们不需要太多的信息就能把它搞清楚。所以，从这个角度，我们可以认为，信息量的度量就等于不确定性的多少。

那么我们如何量化的度量信息量呢？我们来看一个例子，马上要举行世界杯赛了。大家都很关心谁会是冠军。假如我错过了看世界杯，赛后我问一个知道比赛结果的观众"哪支球队是冠军"？ 他不愿意直接告诉我，而要让我猜，并且我每猜一次，他要收一元钱才肯告诉我是否猜对了，那么我需要付给他多少钱才能知道谁是冠军呢? 我可以把球队编上号，从 1 到 32， 然后提问： "冠军的球队在 1-16 号中吗?" 假如他告诉我猜对了， 我会接着问： "冠军在 1-8 号中吗?" 假如他告诉我猜错了， 我自然知道冠军队在 9-16 中。 这样只需要五次，我就能知道哪支球队是冠军。所以，谁是世界杯冠军这条消息的信息量只值五块钱。

当然，香农不是用钱，而是用 "比特"（bit）这个概念来度量信息量。一个比特是一位二进制数，计算机中的一个字节是八个比特。在上面的例子中，这条消息的信息量是五比特。（如果有朝一日有六十四个队进入决赛阶段的比赛，那么"谁世界杯冠军"的信息量就是六比特，因为我们要多猜一次。） 读者可能已经发现, 信息量的比特数和所有可能情况的对数函数 log 有关。 (log32=5, log64=6。）

有些读者此时可能会发现我们实际上可能不需要猜五次就能猜出谁是冠军，因为象巴西、德国、意大利这样的球队得冠军的可能性比日本、美国、韩国等队大的多。因此，我们第一次猜测时不需要把 32 个球队等分成两个组，而可以把少数几个最可能的球队分成一组，把其它队分成另一组。然后我们猜冠军球队是否在那几只热门队中。我们重复这样的过程，根据夺冠概率对剩下的候选球队分组，直到找到冠军队。这样，我们也许三次或四次就猜出结果。因此，当每个球队夺冠的可能性（概率）不等时，"谁世界杯冠军"的信息量的信息量比五比特少。香农指出，它的准确信息量应该是

= -（p1*log p1 + p2 * log p2 + ．．． ＋p32 *log p32),

其中，p1，p2， ．．．，p32 分别是这 32 个球队夺冠的概率。香农把它称为"信息熵" (Entropy)，一般用符号 H 表示，单位是比特。有兴趣的读者可以推算一下当 32 个球队夺冠概率相同时，对应的信息熵等于五比特。有数学基础的读者还可以证明上面公式的值不可能大于五。对于任意一个随机变量 X（比如得冠军的球队），它的熵定义如下：

变量的不确定性越大，熵也就越大，把它搞清楚所需要的信息量也就越大。

有了"熵"这个概念，我们就可以回答本文开始提出的问题，即一本五十万字的中文书平均有多少信息量。我们知道常用的汉字（一级二级国标）大约有 7000 字。假如每个字等概率，那么我们大约需要 13 个比特（即 13 位二进制数）表示一个汉字。但汉字的使用是不平衡的。实际上，前 10% 的汉字占文本的 95% 以上。因此，即使不考虑上下文的相关性，而只考虑每个汉字的独立的概率，那么，每个汉字的信息熵大约也只有 8-9 个比特。如果我们再考虑上下文相关性，每个汉字的信息熵只有5比特左右。所以，一本五十万字的中文书，信息量大约是 250 万比特。如果用一个好的算法压缩一下，整本书可以存成一个 320KB 的文件。如果我们直接用两字节的国标编码存储这本书，大约需要 1MB 大小，是压缩文件的三倍。这两个数量的差距，在信息论中称作"冗余度"（redundancy)。 需要指出的是我们这里讲的 250 万比特是个平均数，同样长度的书，所含的信息量可以差很多。如果一本书重复的内容很多，它的信息量就小，冗余度就大。