Beauty of Mathematics Notebook

Haomin XU

第1章：文字和语言 vs 数字和信息

- 通信等原理和信息传播的模型

- （信源）编码和最短编码

  在罗马体系的文字中，常用字短，生僻字长。在意型文字中，也是类似大都常用字笔画少，生僻字笔画多。

- 解码的规则，语法

  从字母到词的构词法（Morphology）是词的编码规则，语法则是语言的编码和解码规则。

  不过相比较而言，词可以被认为是有限而且封闭的集合，而语言则是无限和开放的集合。

  封闭集合可以有完备的编码解码规则。

- 聚类

  古埃及的象形文字中，读音相同的词可能用同一种符号记录。

- 校验位

  犹太学者抄写《圣经》的时候，每一个希伯来字母对应一个数字，这样每行文字加起来便得到一个特殊的数字。这个数字便成了这一行的校验码。如果某行的校验码和原文中对不上，则说明这行中至少有一个抄写错误。

- 双语对照文本，语料库和机器翻译

  1. 信息的冗余是信息安全的保障。（罗赛塔石碑上的内容是同一信息用不同语言重复三次）
  2. 语言的数据（语料），尤其是双语或多语短对照语料对翻译至关重要，是机器翻译研究的基础。

- 多义性和利用上下文消除歧义性

文字按照意思来类聚，最终会带来一些奇异性。利用上下文来消除部分歧义（Disambiguation）。

第2章：自然语言处理

- Alan Turing最早提出一种验证机器是否有智能的方法：让人和机器进行交流，如果人无法判断自己交流的对象是人还是机器，就说明这个机器有智能了。（图灵测试 Turing Test）
- 误区

  20世纪50年代到70年代，计算机处理自然语言局限在人类学习语言的方式上，用电脑模拟人脑。「计算机理解自然语言」，分析语句和获取语义。

  局限性：

  1. 文法规则的数量大，来不及人工制定，甚至会出现矛盾。
  2. 自然语言和文法和计算机高级程序语言的文法不同。自然语言有词义和上下文 相关的特性（Context Dependent Grammar）。而程序语言是人为设计的，便于计算机解码的上文无关文法（Context Independent Grammar）。

     利用计算复杂度（Computational Complexity）来衡量算法的耗时。对于上下文无关文法，算法的复杂度基本上是语句长度的2次方；上下文有关文法则是6次方。
- 从规则到统计

  1970年后，统计语言学的出现使得自然语言处理重获新生。

第3章：统计语言模型

计算机处理自然语言，一个基本问题就是为自然语言这种上下文相关的特性建立数学模型（统计语言模型 Statistical Language Model）。

1. 用数学的方法描述语言规律

   一个句子是否合理，就看它的可能性大小如何。假定S表示某一个有意义的句子，有一连串特定顺序排列的词$w_1, w2, \ldots, w_n$组成，n为句子的长度。S在文本中出现的可能性为$P(S)$。

$$P(S) = P(w_1, w2, \ldots, w_n)$$
$$= P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_1, w_2) \cdots P(w_n|w_1, w2, \ldots, w_{n-1})$$

   $P(w_n|w_1, w2, \ldots, w_{n-1})$的可能性太多无法，利用马尔可夫(Andrey Markov)假设进行估算。

$$P(S) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_2) \cdots P(w_n|w_{n-1}) \qquad (3.3)$$

   在公式(3.3)对应的统计语言模型是二元模型（Bigram Model）。一个词由前面$N-1$个词决定，对应的模型为N元模型。

   由联合概率$P(w_{i-1}, w_i)$和边缘概率$P(w_{i-1})$估计条件概率：

$$P(w_i|w_{i-1}) = \frac{P(w_{i-1}, w_i)}{P(w_{i-1})}$$

根据相对频度和大数定理，条件概率$P(w_i|w_{i-1})$可估计为：

$$P(w_i|w_{i-1}) \approx \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})}$$

$\#(w_{i-1}, w_i)$理解为w_{i-1}, w_i这对词统计在文本（*语料库 Corpus*）中前后相邻出现的次数。

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更有效。

2. 统计语言模型的工程诀窍
   - 高阶语言模型

     假定文本中的每个词w_i和前面$N-1$个词有关，而与更前面的词无关，则

     $$P(w_i|w_1, w2, \ldots, w_{i-1}) = P(w_i|w_{i-n+1}, w_{i-n+2}, \cdots, w_{i-1}) \qquad (3.10)$$

     这种假设称为$N-1$阶马尔可夫假设，对应的语言模型称为N元模型*(N-Gram Model)*。$N=1$的一元模型实际上是一个上下文无关的模型。而实际应用中最多的是$N=3$的三元模型。当模型从3到4乃至更高元，效果的提升不明显，而资源的耗费却增加得非常快。

   - 模型的训练、零概率问题和平滑方法

     使用语言模型需要知道模型中所有的条件概率（模型的参数）。通过对语料的统计，得到这些参数的过程称作模型的训练。

     如何正确地训练一个语言模型?

     一是增加数据量，因为有大数定理（*Law of Large Numbers*），所以足够多的观测值可以估计模型的参数。

     二是即使有足够多数据量仍然会出现的零概率或统计量不足的问题。训练统计语言模型就在于解决好统计样本不足时的概率估计问题。古德-图灵估计（*Good-Turing Estimate*）给出了这种情况下的估算概率的方法：对于未见的事件，不能认为发生的概率为零，因此从概率的总量（*Probability Mass*）中分配一个很小的比例给未见的事件。 需要将所有可见事件概率根据“越是不可信的统计折扣越多”的方式调小。

     假定预料库中出现r次的词有N_r个，未出现的词量数为N_0，语料库大小为N。

     $$N = \sum_{r=1}^{\infty} r N_r$$

     出现r次的词在整个语料库中的相对频度（*Relative Frequency*）是rN_r/N。以相对平度作为这些词的概率估计。

     根据古德-图灵估计，当r比较小时，需要用更小的概率d_r估计出现r次的词的概率。

     $$d_r = (r+1) \cdot N_{r+1}/N_r < r \cdot N_r/N_r$$
     $$\sum_r d_r \cdot N_r = N$$

     根据Zipf's Law，r越大，词的数量N_r越小，即$N_{r+1} < N_r$。在实际的自然语言处理中，一般对出现次数超过某个阈值的词，频率不下调，只对出现次数低于阈值的词下调频率。下调得到的频率总和给未出现的词。

     Example：对于二元组(w_{i-1}, w_i)的条件概率估计$P(w_i|w_{i-1})$做古德-图灵估计

$$P(w_i|w_{i-1}) = \begin{cases} f(w_i|w_{i-1}) & if\#(w_{i-1},w_i) \geq T \\ f_gt(w_i|w_{i-1}) & if0 < \#(w_{i-1},w_i) < T \\ Q(w_{i-1}) \cdot f(w_i) & otherwise \end{cases}$$

其中T是一个阈值，一般在8-10左右，函数$f_{gt}()$表示经过古德-图灵估计后的相对频度。

$Q(w_i - 1) = \frac{1-\sum_{w_i seen} P(w_i|w_{i-1})}{\sum_{w_i unseen} f(w_i)}$

- 语料的选取问题
  - 训练语料和模型应用的领域需相关
  - 训练数据越多越好
  - 对训练数据进行预处理

第4章：谈谈分词

词是表达语义的最小单位。对于西方拼音语言，词之间有明确的分界符(Delimit)。而一些亚洲语言（如中文），词之间没有明确的分界符。需要先对句子进行分词，才能做进一步的自然语言处理。

利用统计语言模型分词的方法，假定一个句子有几种分词方法，那么最好的一种分词方法应该保证分词完成后这个句子出现的概率最大。

实现技巧：动态规划（Dynamic Programming），维特比（Viterbi）算法。

- 分词的一致性

  简单依靠与人工分词的结果比较来衡量分词器等准确性很难且无意义。因为不同人对于分词的看法还有差异。

- 词的颗粒度和层次

  在机器翻译中，一般颗粒度大翻译效果好；在网页搜索中，小的颗粒度比大的颗粒度好。

  好的做法是，让一个分词器同时支持不同层次的词的分词：首先需要一个基本词表和一个复合词表。然后根据基本词表和复合词表各建立一个语言模型，如L1和L2。再分别根据词表和语言模型进行分词。（这样做保证的是即使在词表和语言模型两个数据库变化的情况下，分词器程序的相同）

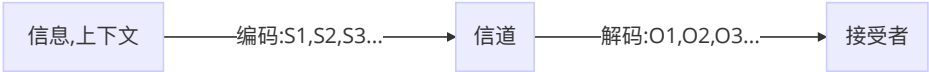
- 分词器准确性

  分词的不一致性可以分为错误和颗粒度不一致。错误又分为越界型和覆盖型。

  近来中文分词工作的重点是做数据挖掘，不断完善复合词的词典。

第5章：隐含马尔可夫模型

1. 通信模型



$s_1, s_2, s_3, \cdots$表示信息源发出的信号。$o_1, o_2, o_3, \cdots$表示接收器接收到的信号。根据接收端的观测信号来推测信号源发生的信息，需要从所有的源信息中找到最可能产生出观测信号的那个信息。即：

$$s_1, s_2, s_3, \cdots = Arg_{all\ s_1, s_2, s_3, \ldots} MaxP(s_1, s_2, s_3, \cdots | o_1, o_2, o_3, \cdots)$$

Arg表示Argument，能获得最大值的那个信息串。利用贝叶斯公式，间接计算上面的概率：

$$\frac{P(o_1, o_2, o_3, \cdots | s_1, s_2, s_3, \cdots) \cdot P(s_1, s_2, s_3, \cdots)}{P(o_1, o_2, o_3, \cdots)}$$

一旦信息$o_1, o_2, o_3, \cdots$产生了就不会改变，因此$P(o_1, o_2, o_3, \cdots)$就是一个可以忽略的常数。上述公式可以等价为：

$$P(o_1, o_2, o_3, \cdots | s_1, s_2, s_3, \cdots) \cdot P(s_1, s_2, s_3, \cdots)$$

这个公式可以用隐含马尔可夫模型 *（Hidden Markov Model）*来估计。

2. 隐含马尔可夫模型
   - 马尔可夫链

     从相对静态的随机变量到随机变量到时间序列$s_1, s_2, s_3, \cdots, s_t, \cdots$，即随机过程（动态）。首先，在任意一个时刻$t$，对应的状态$s_t$都是随机的。第二，任一状态$s_t$的取值都可能和周围其他的状态相关。马尔可夫提出一种简化假设，即随机过程中各个状态s_t的概率分布只与它的前一个状态s_{t-1}有关，即$P(s_t | s_1, s_2, s_3, \cdots, s_t - 1) = P(s_t | s_{t-1})$。符合这个假设的随机过程被称为马尔可夫过程，或马尔可夫链。
   - 隐含马尔可夫模型

     任意时刻t的状态s_t不可见。观察者无法通过观察一个状态序列$s_1, s_2, s_3, \cdots, s_t$来推测转移概率等参数。但隐含马尔可夫模型在每个时刻t会输出一个符号，o_t且o_t仅跟s_t相关（独立输出假设）。基于马尔可夫假设和独立输出假设，某个特定状态序列$s_1, s_2, s_3, \cdots$产生输出符号$o_1, o_2, o_3, \cdots$的概率为：

     $$P(s_1, s_2, s_3, \cdots, o_1, o_2, o_3, \cdots) = \prod_t P(s_t | s_{t-1}) \cdot P(o_t | s_t)$$

     将马尔可夫假设和独立输出假设应用：

     $$P(o_1, o_2, o_3, \cdots | s_1, s_2, s_3, \cdots) = \prod_t P(o_t | s_t)$$

     $$P(s_1, s_2, s_3, \cdots) = \prod_t P(s_t | s_{t-1})$$

     在公式中$P(s_1, s_2, s_3, \cdots)$是语言模型。转移概率 *（Transition Probability）*：$P(s_t | s_{t-1})$。生成概率 *（Generation Probability）*：$P(o_t | s_t)$。

   - 隐含马尔可夫的三个基本问题

1. 给定一个模型，如何计算某个特定的输出序列的概率。

   Forward-Backward算法

2. 给定一个模型和某个特定的输出序列，如何找到最可能产生这个输出的状态序列。

   维特比算法

3. 给定足够量的观测数据，如何估计隐含马尔可夫模型的参数。

   模型训练，计算或估算转移概率和生成概率。

   方法1，有监督的训练方法（*Supervised Training*），前提需要大量人工标记（*Human Annotated*）数据。

   方法2，无监督的训练方法，仅通过大量观测到的信号$o_1, o_2, o_3, \cdots$就能推算模型参数转移概率和生成概率。主要使用鲍姆-韦尔奇算法（*Baum-Welch Algorithm*）：

   - 不同的隐含马尔可夫模型可以产生相同的输出信号。
   - 首先找到一组能够产生输出序列O的模型参数作为初始模型$M_{\theta 0}$。
   - 假定解决了第一个和第二个问题，可以计算出这个模型产生O的概率$P(O|M_{\theta 0})$，还可找到这个模型产生O的所有可能路径以及这些路径的概率。可将其看作是“标注的训练数据”，利用

     $$P(o_t|s_t) \approx \frac{\#(o_t, s_t)}{\#(s_t)} \quad \text{足够多人工标记数据统计估算（有监督训练）}$$

     $$P(s_t|s_{t-1}) \approx \frac{\#(s_{t-1}, s_t)}{\#(s_{t-1})} \quad \text{同统计语言模型的训练方法} P(w_i|w_{i-1}) \approx \frac{\#(w_{i-1}, w_i}{\#(w_{i-1})}$$

     计算出一组新的模型参数$M_{\theta 1}$,使得

     $$P(O|M_{\theta 1}) > P(O|M_{\theta 0})$$

   - 之后不断迭代，直到模型质量不再有明显提升。

   鲍姆-韦尔奇算法每次迭代是不断估计使得输出的概率最大化（*Expectation Maximization*），简称EM过程。EM过程只能保证算法一定能收敛到局部最优解。有监督的训练可以收敛到全局最优点。

使用隐含马尔可夫模型包括一个训练算法（鲍姆-韦尔奇算法）和使用时的解码算法（维特比算法）。

第6章：信息的度量和作用

1. 信息熵

   信息量就等于不确定性的多少。“比特”（Bit）来度量信息量，信息量的比特数和所有可能情况的对数函数log_2有关。熵的定义如下：

   $$H(X) = -\sum_{x \in X} P(x) log_2 P(x)$$

   变量的不确定性越大，熵也越大，所需的信息量就越多。信息熵具有三条性质：

   - 单调性，发生概率越高的事件，其所携带的信息熵越低。

- 非负性，信息熵不能为负。
- 累加性，多随机事件同时发生存在的总不确定性的度量可以表示为各事件不确定性的度量和：

$$
\begin{aligned}
if,\ \ P(A,B) = P(A)\cdot P(B) \\
then,\ H(A,B) = H(A) + H(B)
\end{aligned}
$$

引申：霍夫曼编码（为出现概率高的字符分配短码）

2. 信息的作用

   信息是消除系统不确定性的唯一办法。自然语言处理的大量问题就是寻找相关的信息消除不确定性。

   假定X和Y是两个随机变量，定义在Y的条件下的条件熵（Conditional Entropy）为：

$$
\begin{aligned}
H(X|Y) = -\sum_{x\in X, y\in Y} P(x,y) log_2 P(x|y) \\
H(X) \geq H(X|Y)
\end{aligned}
$$

3. 互信息

   假定又两个随机事件X和Y，它们的互信息定义如下：

$$
\begin{aligned}
I(X;Y) = \sum_{x\in X, y\in Y} P(x,y) log_2 \frac{P(x,y)}{P(x)P(y)} \\
I(X;Y) = H(X) - H(X|Y)
\end{aligned}
$$

   两个事件相关性的量化度量就是在了解其中一个Y的前提下，对消除另一个事件X不确定性所提供的信息量。互信息是取值在0到$min(H(X), H(Y))$之间的函数。当X和Y完全相关时，$I(X,Y) = H(X)$，且$H(X) = H(Y)$；当两者无关时，$I(X,Y) = 0$。

   使用互信息可以有效地解决词义的二义性。

4. 相对熵（Relative Entropy or Kullback-Leibler Divergence）

   和互信息不同，相对熵用来衡量两个取值为正数的函数的相似性：

$$
KL(f(x)||g(x)) = \sum_{x\in X} f(x)\cdot log\frac{f(x)}{g(x)}
$$

   - 对于两个完全相同的函数，它们的相对熵等于零。
   - 相对熵越大，两个函数差异越大；相对熵越小，两个函数差异越小。
   - 对于概率分布或者概率密度函数，如果取值均大于零，相对熵可以度量两个随机分布的差异性。
   - 相对熵是不对称的：

$$
KL(f(x)||g(x)) \neq KL(g(x)||f(x))
$$

詹森和香农提出新的相对熵计算方法：

$$JS(f(x)||g(x)) = \frac{1}{2}[KL(f(x)||g(x)) + KL(g(x)||f(x))]$$

相对熵的应用：

- 衡量两段信息的相似程度，相对熵越小，说明两段信息越接近。
- 衡量两个常用词在不同文本中的概率分布，看它们是否同义。
- 词频率-逆向文档频率(TF-IDF)。

5. 小结

   信息熵能直接用于衡量统计语言模型的好坏。

   对高阶的语言模型，应该用条件熵。

   如从训练语料和真实应用的文本中得到的概率函数有偏差，需要再引入相对熵。

   语言模型负载度（Perplexity）：在给定上下文的条件下，句子中每个位置平均可以选择的单词数量。一个模型的复杂度越小，每个位置的词就越确定，模型越好。

第7章：贾里尼克和现代语言处理

- “巨人”的力量
- 将统计方法应用于自然语言处理的必然
- IBM最早提出统计语音识别和自然语言处理的历史必然性：计算能力、语料和人。

第8章：简单之美，布尔代数和搜索引擎

建立一个搜索引擎需要做的是：自动下载尽可能多的网页；建立快速有效的索引；根据相关性对网页进行公平准则的排序。（下载，索引，排序）

索引在搜索引擎中最基础也最为重要。

1. 布尔代数

   运算元素：1和0 。基本运算：与(AND)，或(OR)，非(NOT) 。

   布尔运算与文献检索的关系，对于一个用户输入的关键词，搜索引擎要判断每篇文献是否含有这个关键词，从而对每篇文献分配一个逻辑值。

2. 索引

图书馆索引卡片类比互联网搜索引擎。

最简单的索引结构：有多少文献就用多少位数对应。1代表相应文献有这个关键字，0代表没有。

互联网搜素引擎，每个网页就是一个文献。索引量巨大，在万亿字节的量级。索引需要通过分布式的方式存储到不同服务器上。普遍做法是根据网页序号将索引分成很多份（Shards），分别存储在不同的服务器中，服务器并行处理用户请求。

第9章：图论和网络爬虫

离散数学四大分支：数理逻辑（基于布尔运算），集合论，图论，近世代数。

如何自动下载互联网网页？（图论的遍历Traverse算法）

1. 图论

   一种图的遍历算法“广度优先搜索”（Breadth-First Search，BFS），尽可能广地访问与每个节点直接链接的其他节点。 It starts at the tree root and explores the neighbor nodes first, before moving to the next level neighbours.

   另一种图的遍历算法“深度优先搜索”（Depth-First Search，DFS）,"一路走到黑"。One starts at the root and explores as far as possible along each branch before backtracking.

2. 网络爬虫（Web Crawlers）

   互联网看作图：每个网页是一个节点，每个Hyperlinks当作连接网页的弧。

   网络爬虫：从任何一个网页出发，用图的遍历算法，自动访问每个网页并把它们存起来。在网络爬虫中，使用“散列表”(Hash Table)来记录网页是否下载过，避免重复下载。

3. 图论的两点补充

   3.1 欧拉七巧问题

   图中每一个顶点，与之相连的边的数量称为它的度（Degree）。

   定理：如果一个图能够从一个顶点出发，每条边不重复地遍历一遍回到这个顶点，那每一顶点的度必须是偶数。

   3.2 构建网络爬虫的工程要点

   - BFS还是DFS

     在不考虑时间因素，互联网静态不变的情况下，这两个算法的时间复杂度大概相同「$O(V+E)$是节点数量V和边的数量E之和的线性函数」

     现实网络爬虫问题应定义为：如何在有限的时间内最多地爬下最重要的网页。这时显然BFS优于DFS。

     涉及到爬虫的分布式结构和网络通信的握手（下载服务器和网站的服务器建立通信的过程）成本：握手次数过多，下载效率下降。网络爬虫是由成千上万服务器组成的分布系统。对于一个网站需要特定的服务器先下载完，再下载别的网站(DFS)。

     总结，网络爬虫对网页遍历不是简单的BFS或DFS，而是有相对复杂的下载优先级排序方法。由调度系统（Scheduler）管理。

   - 页面分析和URL的提取

     一个网页下载完，需要提取网页中的URL，再加入下载队列中。

早期网页直接用HTML编写，URL方便提取。现在很多使用脚本语言（如JS）生成，解析困难，需要模拟浏览器运行一个网页（解析），然后得到隐含的URL。

- 记录已下载过的网页——URL表

  为了防止一个网页（节点）被下载多次，用散列表记录已下载过的网页。

  上千台服务器一起下载网页，维护散列表很困难。这张表可能会大到一台服务器存不下；每个下载服务器在下载前和完成后都要访问和维护散列表，以免不同服务器重复工作，对存储散列表的服务器的通信成了整个爬虫系统的瓶颈。

  解决方法：

  - 明确每台下载服务器的分工，即调度时一个URL的下载任务对应一个服务器，以免重复判断。
  - 在明确分工基础上，批处理判断URL是否下载。（如每次向散列表发送一大批询问，或者每次更新一大批散列表内容）

第10章：PageRank（Google的民主表决式网页排名技术）

搜索结果的排名取决于两组信息：关于网页质量信息（Quality）和这个查询与每个网页的相关性信息（Relevance）。

1. PageRank算法原理（衡量网页质量）

   PageRank的核心思想：如果一个网页被很多其他网页所链接，说明它受到普遍的承认和信赖，它的搜索排名就高。且网页排名高的网站贡献的链接权重大。

   - 网页本身的排名决定了其指向其他网页的权重，指向同一网页的权重相加为该网页的排名。
   - 二维矩阵相乘，假定所有网页排名相同，根据初始值不断迭代排名。
   - 二维矩阵理论上有网页数量平方多个元素，稀疏矩阵计算技巧简化计算量。MapReduce工具自动化PageRank并行计算。
   - 今天决定搜索质量最有用的信息是用户的点击数据。

2. PageRank的计算方法

   假定向量$B=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$为第一，第二，$\cdots$，第N个网页的排名。矩阵

   $A=\begin{bmatrix} a_{11},\cdots,a_{1n},\cdots,a_{1M} \\ \cdots \\ a_{m1},\cdots,a_{mn},\cdots,a_{mM} \\ \cdots \\ a_{M1},\cdots,a_{Mn},\cdots,a_{MM} \end{bmatrix}$为网页之间链接的数目，其中$a_{mn}$代表第$m$个网页指向第$n$个网页的链接数量。$A$是已知，$B$是未知的。

   初始假设，所有网页的排名都是1/N，即$B_0=(\frac{1}{N},\frac{1}{N},\cdots,\frac{1}{N})$。

   B_i是第i次迭代的结果：$B_i=A\cdot B_{i-1}$。B_i最终会收敛，即B_i无限接近于B，$B=B\times A$。此时停止迭代，算法结束。一般迭代10次基本就收敛了。

由于网页之间链接数量相比互联网的规模非常稀疏，因此计算网页的排名也需要对零概率事件进行平滑处理。即$B_i = [\frac{\alpha}{N} \cdot I + (1-\alpha)A] \cdot B_{i-1}$，其中N是互联网网页数量，$\alpha$是一个较小的常数，$I$是单位矩阵。

第11章：如何确定网页和查询的相关性

影响搜索引擎质量的因素：

- 完备的索引
- 对网页质量的度量，如PageRank，对网页权威性的衡量等。
- 用户偏好
- 确定一个网页和某个查询的相关性的方法。

1. 搜索关键词权重的科学度量TF-IDF

   根据网页长度，对关键词的次数进行归一化，也就是用关键词的次数除以网页总字数，称为关键词的频率或单文本词频（*Term Frequency*）。简单度量网页和查询的相关性的方法，将各相关关键词的权重相加。即对于查询关键词$w_1, w_2, \cdots, w_N$，在特定网页中词频为$TF_1, TF_2, \cdots, TF_N$,则这个查询和该网页点相关性为：$TF_1 + TF_2 + \cdots + TF_N$。

   停止词（Stop Word）如“的”、“是”、“和”，度量相关性不应考虑其频率。

   需对汉语中每个词分配一个权重：一个词的预测主题的能力越强，权重越大，反之越小；停止词的权重为零。假定一个关键词w在D_w个网页中出现过，D_w越大，w的权重就越小。逆文本频率指数（*Inverse Document Frequency, IDF*）。

   $$IDF = log_2(\frac{D}{D_w})$$

   D为全部的网页数。查询和该网页的相关性为加权求和：

   $$TF_1 \cdot IDF_1 + TF_2 \cdot IDF_2 + \cdots + TF_N \cdot IDF_N$$

2. TF-IDF的信息论依据

   TF-IDF的概念是特定条件下关键词的概率分布的交叉熵（相对熵KLD）。

   查询词w的信息量为：

   $$\begin{aligned} I(w) &= -P(w)logP(w) \\ &= -\frac{TF(w)}{N} log_2 \frac{TF(w)}{N} \\ &= \frac{TF(w)}{N} log_2 \frac{N}{TF(w)} \end{aligned}$$

   其中N是整个语料库的大小，是个可以省略的常数，上面公式简化为：

   $$I(w) = TF(w) log_2 \frac{N}{TF(w)}$$

   缺陷：两个词出现的频率TF相同，一个是某特定文章的常见词（有更高分辨率，权重应该更大），另一个是分散在多篇文章的词。

$$TF-IDF(W)=TF(w)log_2\frac{D}{D(w)}=I(w)-TF(w)log_2\frac{M}{c(w)}$$

其中，M为每个文献的词数（假设相同），$M=\frac{N}{D}=\frac{\sum_w TF(w)}{D}$。$c(w)=\frac{TF(w)}{D(w)}$表示一个词在一个文献中出现的次数，$c(W)<M$。一个词的信息量$I(w)$越多，TF-IDF越大；且含有$w$的文献中$w$出现的平均次数越多，$c(w)$越大，TF-IDF也越大。

第12章：有限状态机和动态规划（地图与本地搜索的核心技术）

智能手机的定位和导航功能，有三项关键技术：

- 利用卫星定位
- 地址的识别
- 根据用户输入的起点和终点规划最优路线

1. 地址分析和有限状态机

   地址大描述文法依然是复杂的上下文有关文法。有效的地址分析方法是有限状态机。有限状态机是一个特殊的有向图，包括一些状态（节点）和连接这些状态的有向弧。

   使用有限状态机识别地址，关键需要解决两个问题：通过一些有效的地址建立状态机；给定一个有限状态机后，地址字串的匹配算法。

   为了解决“严格匹配”的问题，提出模糊匹配，基于概率的有限状态机（和离散马尔可夫链基本等效）。

2. 全球导航和动态规划

   关键算法是计算机科学图论中的动态规划（Dynamic Programming）。减少计算复杂度。

   寻找最短路径，将寻找全程最短路线问题分解成一个个寻找局部最短路线的小问题。

3. 有限状态传感器

   有限状态机严格的数学定义：有限状态机是一个五元组$(\Sigma, S, s_0, \delta, f)$。

   Σ: 输入符号的集合

   S: 一个非空的有限状态集合

   s_0: S中的一个特殊状态，起始状态

   δ: 从空间$S\times\Sigma$到S的映射函数，$\delta: S\times\Sigma\rightarrow S$

   f: S中的一个特殊状态，终止状态

   在语音识别和自然语言理解中使用特殊的有限状态机：加权的有限状态传感器（Weighted Finite State Transducer, WFST）。（状态机中的每个状态由输入和输出符号定义）

第13章：Google AK-47的设计者（阿米特·辛格博士）

一个好的算法应该像AK-47冲锋枪一样，简单、有效、可靠性好且容易读懂。

第14章：余弦定理和新闻的分类

计算机自动整理、分类和聚类各个新闻网站的内容。

1. 新闻的特征向量

   新闻的分类，要求先把文字的新闻变成一组可计算的数字，然后再设计一个算法来算出任意两篇新闻的相似性。对于一片新闻中所有的实词，计算出它们的TF-IDF值，再把这些值按照对应的实词在词汇表的位置依次排序，得到一个N维向量。这个向量称为新闻的特征向量（Feature Vector）。

2. 向量距离的度量

   如果两篇新闻属于同一类，它们的特征向量在某几个维度的值都比较大。文本长度不同，特征向量各个维度的数值也不同，因此可以通过计算两个向量的夹角来判断对应的新闻主题的接近程度。

   $$cosA = \frac{<b,c>}{|b||c|}$$

   其中，分母表示两个向量$\vec{b}$和$\vec{c}$的长度，分子表示两个向量的内积。夹角的余弦越小，夹角越大，说明两篇新闻的相关性越小。

   新闻分类算法：

   第一种对于已知特征向量的新闻，容易计算需要被分类新闻与之地余弦相似性。

   第二种，如果事先没有这些新闻类别的特征向量。“自底向上不断合并”的方法：

   I. 计算所有新闻之间两两的余弦相似性，把相似性大于一个阈值的新闻合成一个小类（Subclass）。N篇新闻就被合并成N_1个小类，$N_1 < N$。

   II. 每个小类中所有新闻作为一个整体，计算小类的特征向量。再计算小类之间两两点余弦相似性，然后合并成N_2大类，$N_2 < N_1$。

   III. 当某一大类里新闻之间的相似性小了，就停止上述迭代过程。

3. 计算向量余弦的技巧

   大数据量时的余弦计算。如果不简化算法，比较所有N篇新闻之间两两的相关性进行一次迭代的计算复杂度为$O(N^2 \cdot |a|)$。需要对其进行简化：

   - 计算两个向量余弦时，分母不需重复计算，即可以将长度存起来，需要时直接取用。
   - 计算分子即两个向量内积时，只需考虑向量中的非零元素。
   - 删除虚词，如“的”、“是”、“因为”、“所以”等。

   位置的加权。出现在文本不同位置的词在分类时的重要性也不同。因此需要对标题和重要位置的词进行额外的加权，以提高文本分类的准确性。

第15章：矩阵运算和文本处理中的两个分类问题

1. 文本和词汇的矩阵

自然语言处理中，最常见的两个分类问题：将文本按主题归类和将词汇表中的字词按意思归类。一步到位的方法：矩阵运算中的奇异值分解（Singular Value Decomposition, SVD）。

矩阵A描述文章和词的关联性，每一行对应一篇文章，每一列对应一个词：

$$A = \begin{bmatrix} a_{11}, \cdots, a_{1j}, \cdots, a_{1N} \\ \cdots \\ a_{i1}, \cdots, a_{ij}, \cdots, a_{iN} \\ \cdots \\ a_{M1}, \cdots, a_{Mj}, \cdots, a_{MN} \end{bmatrix}$$

$M \times N$的矩阵对应M篇文章和N个词。a_{ij}表示第j个词在第i篇文章中出现的加权词频（如TF-IDF值）。

这个关联性矩阵非常庞大，需要对其进行奇异值分解。假设$M = 1000000, N = 500000$:

$$\begin{bmatrix} \cdots\cdots \\ \cdots\cdots \\ A \\ \cdots\cdots \\ \cdots\cdots \end{bmatrix}_{1000000 \times 500000} = \begin{bmatrix} \cdots \\ \cdots \\ X \\ \cdots \\ \cdots \end{bmatrix}_{1000000 \times 100} [B]_{100 \times 100} [\cdots\cdots Y \cdots\cdots]_{100 \times 500000}$$

矩阵X（文本x主题）表示对文本分类的结果；矩阵Y（语义类x词）表示对词进行分类的结果；B（主题x语义类）表示词的类和文章的类之间的相关性。（？书中的描述反了）

2. 奇异值分解的方法和应用场景

$$A_{MN} = X_{MM} \times B_{MN} \times Y_{NN}$$

其中X是一个酉矩阵（Unitary Matrix），Y是一个酉矩阵的共轭矩阵。与其共轭矩阵转置相乘等于单位阵的矩阵为酉矩阵。B是一个对角矩阵。

奇异值分解分两步：第一步，将矩阵A变成一个双对角矩阵，计算复杂度为$O(MN^2), M > N$。第二步，将双对角矩阵变成奇异值分解的三个矩阵，这步计算量小。

比较利用余弦定理计算文本相似度，它们一次迭代的计算复杂度处于一个数量级，但奇异值分解不需要多次迭代。奇异值分解也有问题，存储量较大，整个矩阵都需要在内存里计算，而余弦定理聚类则不需要，且利用奇异值分类的结果略粗糙，适合处理超大规模文本粗分类。

应用场景：实际应用中先进行奇异值分解得到粗分类结果，再利用余弦方法在粗分类结果基础上迭代得到精确结果。

第16章：信息指纹及其应用

1. 信息指纹

用一个不太长的随机数作为区别一段信息和其他信息的指纹。例如爬虫中的散列表，计算并存储对应的信息指纹比存储URL字符串节省空间，且对于整数的查找比字符串快。

网址（字符串）计算信息指纹的方法：第一步，将字符串看成是特殊的长整数（计算机中字符按照整数来存储）。第二步，用伪随机数产生器算法（Pseudo-Random Number Generator, PRNG），将任意很长的整数转换成特定长度的伪随机数。(现常用梅森旋转算法，Mersenne Twister)

信息指纹的一个特征是不可逆性，无法通过信息指纹推出原有信息。互联网加密使用基于加密的伪随机产生器（Cryptographically Secure Pseudo-Random Number Generator, CSPRNG）。常用算法有MD5和SHA-1等标准。

2. 信息指纹的用途

   2.1 集合相同的判定

   - 集合中元素意义比较，时间复杂度$O(N^2)$
   - 将两个集合元素分别排序，然后顺序比较。$O(Nlog_2N)$
   - 将第一个集合放在散列表中，然后将第二个集合元素一一和散列表比较。时间复杂度$O(N)$，额外使用N的空间，且代码复杂。
   - 计算两个集合的指纹直接比较。

     $$FP(S) = FP(e_1) + FP(e_2) + \cdots + FP(e_n)$$

     加法保证了几何的指纹不受次序干扰。指纹相同，则两个集合相同。

   2.2 判定集合基本相同

   具体做法是选取并比较文本或网页中特征集合（IDF值大的）的信息指纹。

   2.3 YouTube的反盗版

   视频的匹配技术核心：关键帧的提取和特征的提取。

   处理视频图像找到关键帧，再用一组信息指纹来表示这些关键帧。通过比较信息指纹和先后时间顺序来区别原创和盗版。

   广告分成策略：所有视频可插入广告，但广告的全部收益归原视频所有。

3. 信息指纹的重复可能性和相似哈希

   假定一个128位的信息指纹，$N = 2^{128}$。随意挑选两个指纹，重复的可能性为$1/N$，不重复的可能性是$\frac{N-1}{N}$。k个指纹不重复的概率为：

   $$P_k = \frac{(N-1)(N-2)\cdots(N-k+1)}{N^{k-1}}$$

   P_k随k增加而减小，若$P_k < 0.5$，则k个指纹重复一次的数学期望就超过1。估计最大值：

   $$N \to \infty,\ \ P_{k+1} \approx e^{-\frac{1}{n}} e^{-\frac{2}{n}} \cdots e^{-\frac{k}{n}} = exp(-\frac{k(k+1)}{2N}) < 0.5$$

   则$k > \frac{-1+\sqrt{3+8Nln2}}{2}$。对于$N = 2^{128}$，$k > 2^{64}$信息指纹才会重复一次。因此不同信息产生相同指纹的可能性几乎为零。

   相似哈希（Simhash）是一种特殊的信息指纹。（我理解为，用文本中词的信息指纹来推算文本的信息指纹，在收缩这步使其不可逆。）假定网页中有若干词$t_1, t_2, \cdots, t_k$，它们权重（如TF-IDF）分别为$w_1, w_2, \cdots, w_k$。计算相似哈希：

   - 第一步，扩展。将N位的二进制指纹扩展成N个实数。依次在原权重基础上根据每个词的指纹按位加或减去（1或0）每个指纹对应的权重，得到新的N个实数。
   - 第二步，收缩。大于0的实数设置成1，其余为0。得到文本的Simhash。

第17章：谈谈密码学的数学原理

1. 密码学的自发时代

   一种好的编码方法，破译者应该无法从密码中统计处明码的规律，且无法根据已知明文和密文的对应推断出新的密文内容。

2. 信息论时代的密码学

   当密码之间分布均匀并且统计独立时，提供的信息最少。均匀分布使得无法统计，统计独立可以保证即使知道加密算法并且有一段密码和明码对应，也无法破译别的密码。

   RSA算法：

   I. 找到两个很大的素数P和Q，比如100位长的，然后计算乘积。
   $N = P \times Q, M = (P-1) \times (Q-1)$

   II. 找到一个和M互素的整数E（M和E除了1以外没有公约数）

   III. 找一个整数D，使$E \times D \, mode \, M = 1$

   对X进行加密，得到密码Y：

$$X^E \, mode \, N = Y$$

   E为公钥，D为密钥，联系公钥和密钥的乘积N也是公开的。根据费马小定理有：

$$Y^D \, mod \, N = X$$

   RSA公开密钥的好处：

   - 简单，乘除和乘方运算
   - 可靠，保证产生的密文是统计独立而分布均匀的
   - 灵活，可产生很多公开密钥E和私钥D的组合给不同的加密者。

   对于RSA加密方法的破解：

   - 对大字N进行因数分解，即通过N反过来找Q。（GNFS，一般数域筛法）
   - 工程实现或使用不当的漏洞。评论

第18章：谈谈搜索引擎反作弊和搜索结果的权威性问题

搜索引擎给出结果噪音：主要是针对搜索引擎网页排名的作弊（SPAM）；用户在互联网的活动产生（大量不准确信息）

1. 搜索引擎的反作弊

   通信中解决噪音干扰的基本思路适用于搜索反作弊：

   - 从信息源出发，加强通信编码自身的抗干扰能力。
   - 从传输，过滤掉噪音还原信息。

   “术”层面，看到作弊的例子分析并清除：如针对分析作弊的JS跳转页面

   “道”层面，透过具体作弊例子，找到作弊的动机和本质：

   - 针对和商业相关的搜索，采用抗干扰强的算法
   - 对信息类的搜索，采用敏感算法。
   - 针对贩卖链接的网站，计算出链的余弦距离，这些人为作弊的出链的余弦距离几乎为1。

- 图论中，如果有几个节点两两相互都连接在一起，被称为一个Clique。作弊网站一般需要互相链接，可以通过发现Clique反作弊。

2. 搜索结果的权威性

   用户使用搜索引擎一般出于两种目的：导航和查找信息。PageRank和其他关于网页质量的度量方法都很难衡量搜索结果的权威性。

   权威性的度量4步：

   a. 在网页正文中寻找“提及（Mention）”信息。对每个网页正文（包括标题）中的每句话进行句法分析，找出涉及到主题的短语以及对信息源的描述。

   b. 利用互信息，找到主题短语和信息源的相关性。

   c. 对主题短语进行聚合，聚类意思相同的短语。矩阵运算

   d. 对一个网站中的网页进行聚合。比如把网站下的网页按照子域或子目录进行聚类。权威性的度量只能建立在子域或子目录这一级。

   四步完成得到一个针对不同主题的网站权威性关联矩阵。

第19章：谈谈数学模型的重要性

1. 一个正确的数学模型应当在形式上是最简单的。
2. 一个正确的模型一开始可能不如一个精雕细琢国的错误模型准确。但是如果认定大方向是对的，就应该坚持下去。
3. 大量准确的数据对研发很重要。
4. 正确的模型也可能受噪音干扰，这时不应该用凑合的修正方法加以弥补，应找到噪音的根源。

第20章：谈谈最大熵模型

“不要把所有鸡蛋放在一个篮子里”以降低风险→ 最大熵原理（The Maximum Entropy Principle）

1. 最大熵原理和最大熵模型

   最大熵原理指出，对一个随机事件的概率分布进行预测时，预测应满足全部已知条件，而对未知的情况不要做任何主观假设。这种情况下，概率分布信息熵最大，概率分布最均匀，预测风险最小。

   对任何一组不自相矛盾的信息，最大熵模型存在且唯一：

$$P(w_3|w_1,w_2,s)=\frac{1}{Z(w_1,w_2,s)}e^{\lambda_1(w_1,w_2,w_3)+\lambda_2(s,w_3)}$$

   w_3是要预测的词，w_1,w_2是其前两个字，s表示主题。参数Z,λ需要通过训练得到。

2. 最大熵模型的训练
   - 最原始的最大熵模型训练方法：通用迭代算法GIS（Generalized Iterative Scalling）：
     1. 假定第零次迭代的初始模型为等概率的均匀分布。

    2. 用第N次迭代的模型来估算每种信息特征在训练数据中的分布。
    3. 重复2，直到收敛。
  - 改进迭代算法IIS（Improved Iterative Scaling）
  - 最大熵模型快速算法

第21章：拼音输入法的数学原理

输入法输入汉字的快慢取决于汉子编码的平均长度，即击键次数x寻找键的时间。

1. 输入法与编码

   对汉字的编码分为两部分：对拼音的编码（A～Z）和消除歧义性（0～9）的编码。

   全拼输入法的优点：不需要专门学习；输入自然不会中断思维，即找每个键的时间短；编码长，有信息冗余量，容错性好。

2. 输入一个汉字需要敲多少键（香农第一定理）

   香农第一定理：任何一个信息的编码长度都不小于它的信息熵。

   如果对每个字进行统计，且不考虑上下文相关性，大致估算在10比特以内。假定使用26字母输入，每个字母代表$log_2 26 \approx 4.7$比特的信息，即输入一个汉字平均需要$10/4.7 \approx 2.1$次键。

   如果把汉字组成词，再以词为单位统计信息熵，每个汉字的平均信息熵将会减少到大约8比特。再基于词的统计语言模型，每个汉字的信息熵降到6比特左右。输入一个汉字只要敲$6/4.7 \approx 1.3$次。

   利用上下文最好的办法是借助语言模型。借助概率论语言模型可以保证拼音转汉字（解决一音多字问题）的效果最好。

3. 拼音转汉字的算法

   拼音转汉字的算法和在导航中寻找最短路径的算法相同，都是动态规划。

$$
\begin{aligned}
w_1, w2, \cdots, w_N &= Arg_{w \in W} Max P(w_1, w_2, \cdots, w_N | y_1, y_2, \cdots, y_N) \\
&= Arg_{w \in W} Max P(y_1, y_2, \cdots, y_N | w_1, w_2, \cdots, w_N) \cdot P(w_1, w_2, \cdots, w_N) \\
&\approx Arg_{w \in W} Max \prod_{i=1}^{N} P(w_i | w_{i-1}) \cdot P(y_i | w_i)
\end{aligned}
$$

   $y_1, y_2, \cdots, y_N$是使用者输入的拼音串；$w_1, w_2, \cdots, w_N$是对应输入拼音的候选汉字。（w_1表示多个对应y_1的候选汉字）上述连乘取对数同时取反变为累加，寻找最大概率问题变成寻找最短路径问题。可直接利用动态规划算法实现。在拼音转汉字的网格图中，两个节点（词）w_{i-1}和w_i之间的距离是转移概率和生成概率的乘积$-log_2 P(w_i | w_{i-1}) P(y_i | w_i)$。

4. 个性化的语言模型

   速度只是一个而不是唯一的衡量标准，当输入速度超过一定阈值后，用户的体验可能更重要。

   不同人群的用词习惯，说话写作的水平不同，应该有各自的语言模型。

   - 如何训练好一个个性化的语言模型
     1. 将训练语言模型的文本按主题分成不同类，$C_1, C_2, \cdots, C_{1000}$

2. 找到每个类的特征向量（TF-IDF）$X_1, X_2, \cdots, X_{1000}$
3. 统计某人输入的文本，得到他的输入词的特征向量Y
4. 计算Y和$X_1, X_2, \cdots, X_{1000}$的余弦
5. 选择前K个和Y距离最近的类对应的文本，作为这个特定用户的语言模型训练数据
6. 训练一个用户特定的语言模型M_1

- 处理好语言模型M_1和通用语言模型M_0的关系

  综合M_1和M_0，最好采用最大熵模型，但是比较复杂且训练时间长，成本高。采用简化模型：线性插值模型。

$$P'(w_i|w_{i-1}) = \lambda(w_{i-1}) \cdot P_0(w_i|w_{i-1}) + (1 - \lambda(w_{i-1})) \cdot P_1(w_i|w_{i-1})$$

  $\lambda(w_{i-1})$是一个(0,1)之内的插值参数。线性组合P'的熵比P_0和P_1熵的线性组合小，因此新组合模型的不确定性小，是更好的模型。

第22章：自然语言处理教父马库斯和他优秀的弟子们

马库斯的工作证明统计的方法比规则的方法更适合对自然语言做深入的分析。

柯林斯：繁琐哲学，把事情做到极致。

布莱尔：追求简单的方法，观其大略。

第23章：布隆过滤器

1. 布隆过滤器的原理

   背景：软件开发时，经常要判断一个元素是否在一个集合中。一般计算机中的集合是用散列表（Hash Table）来存储的，将每一个元素对应一个多字节的信息指纹。由于散列表的存储效率一般只有50%，当集合规模巨大时候，需要相当大的存储空间。

   布隆过滤器只需要散列表的1/8到1/4点大小就能解决同样问题。

   布隆过滤器（Bloom Filter）实际上是一个很长的二进制向量和一系列随机映射函数。

   - 实现方式：假设存储一亿个邮件地址，先建立一个16亿比特的零向量。对于每个邮件地址X，用8个不同的随机数产生器（$F_1, F_2, \cdots, F_8$）产生8个信息指纹（$f_1, f_2, \cdots, f_8$）。再用一个随机数产生器G把8个信息指纹映射到1-16亿中的8个自然数（$g_1, g_2, \cdots, g_8$），把这8位都设为1。
   - 验证方式：检测一个可以的电子邮件Y是否在黑名单中。用8个随机数产生器（$F_1, F_2, \cdots, F_8$）对这个地址产生8个信息指纹$s_1, s_2, \cdots, s_8$，将8个指纹对应到布隆过滤器的8个比特位$t_1, t_2, \cdots, t_8$，若这8位为1，则为垃圾邮件。
   - 优缺点：布隆过滤器不会错过黑名单中任意一个地址。但是可能将好的地址判定为在黑名单中，误识别率在万分之一以下。常见的补救办法是建立一个小的白名单存储可能被误判的邮件地址。

2. 布隆过滤器的误识别问题

   假定布隆过滤器有m比特，里面有n个元素，每个元素对应k个信息指纹的散列函数。

对于过滤器一个特定的位置，第一个散列函数会把其置1的概率是1/m，它依然为0度概率是$1-\frac{1}{m}$。这个元素的k个散列函数都没有将其置1的概率为$(1-\frac{1}{m})^k$。插入n个元素后，被置1的概率为$1-(1-\frac{1}{m})^{kn}$。

一个不在集合中的元素，所有k个散列函数对应比特位为1度概率为：
$(1-[1-\frac{1}{m}]^{kn})^k \approx (1-e^{-\frac{kn}{m}})^k$.

假定一个元素用16比特（$\frac{m}{n}=16$），k=8，则假阳性的概率为万分之五。

第24章：马尔可夫链的扩展（贝叶斯网络）

1. 贝叶斯网络

   每个状态只跟其直接连接的状态有关，而跟与它间接连接的状态没有直接关系。

   A Bayesian network is probabilistic graphical model that represents a set of variables and their conditional dependencies via a directed acyclic graph(DAG).

   贝叶斯网络的拓扑结构比马尔可夫链灵活，不受链状结构的束缚，可以更准确地描述事件之间的相关性。（马尔可夫链式贝叶斯网络的特例，贝叶斯网络是马尔可夫链的推广）

2. 贝叶斯网络在词分类中的应用

   文章和关键词本身有直接的关联，两者还都和概念有直接关联，同时它们通过主题还有间接的关联。

   Google的Rephil算法。

3. 贝叶斯网络的训练

   得到贝叶斯网络拓扑结构和各个状态之间的相关概率参数的过程分别叫做结构训练和参数训练。

   - 贝叶斯网络结构优化

     优化的贝叶斯网络结构要保证它产生的序列从头到尾的可能性最大。

     - 使用概率做度量（后验概率最大）。计算复杂度NP-Hard
     - 贪心算法（Greedy Algorithm），在每一步沿箭头方向寻找有限步。可能会导致陷入局部最优，远离全局最优解。
     - 蒙特卡罗（Monte Carlo）方法，防止陷入局部最优，计算量较大。
     - 信息论方法，计算节点之间两两对互信息，只保留互信息较大的节点直接的连接，然后再对简化的网络进行完备的搜索找到全局优化结构。

   - 贝叶斯网络参数优化

     确定节点之间的弧度权重。假定权重用条件概率来度量，训练数据使得观察到这些数据的后验概率$P(D|\theta)$最大(EM过程，Expectation Maximization Process)。

   通常结构的训练和参数的训练是交替进行的，直到收敛或误差足够小。

第25章：条件随机场、文法分析及其他

1. 文法分析——计算机算法的演变

自然语言的句法分析（Sentence Parsing）一般指根据文法对一个句子建立分析，建立这个句子的语法树，即文法分析（Syntactic Parsing）；或指对一个句子中各成分的语义进行分析，得到对这个句子语义的描述（如嵌套的框结构，或语义树），即语义分析（Semantic Parsing）。

选择文法规则坚持一个原则：让被分析的句子的语法树概率达到最大。文法分析可以看成是一个括括号的过程：

A1: 是否开始一个新的左括号

A2: 是否继续留在这个括号中

A3: 是否结束一个括号，即标上右括号

每一个括号就是句子的一个成分，括号之间的嵌套关系是不同层面句子成分的构成关系。统计模型$P(A|prefix)$，A表示行动，句子前缀prefix指句子从开头到目前所有词和语法成分。

2. 条件随机场

   大大提高了句子浅层分析的正确率。

   条件随机场是隐含马尔可夫模型的一种扩展（观察值可能与前后状态都有关，无向概率图模型）。

   条件随机场的节点分为状态节点的集合Y和观察变量节点的集合X。整个条件随机场的量化模型就是这两个集合的联合概率分布：

$$P(X,Y)=P(x_1,x_2,\cdots,x_n,y_1,y_2,\cdots,y_m)$$

   由于变量多，不可能得到足够多的数据来用大数定理直接估计。通过边缘分布（Marginal Distribution），来找出符合所有条件并使熵最大的概率分布函数（指数函数）。每个边缘分布，对应指数模型中的一个特征f_i(Feature)。

$$P(x_1,x_2,\cdots,x_n,y_1,y_2,\cdots,y_m)=\frac{e^{f_1+f_2+\ldots+f_k}}{Z}$$

3. 条件随机场在其他领域的应用

   犯罪发生概率的条件随机场模型：$P(l,t,c,X)$。l,t,c,X分别表示地点，时间，犯罪类型以及天气活动等因素的合集X。

$$P(l,t,c|X)=\frac{P(l,t,c,X)}{P(X)}$$

   $P(X)$由历史数据推算，可以看作已知。$P(l,t,c,X)$则用条件随机场估算。

第26章：维特比和他的维特比算法

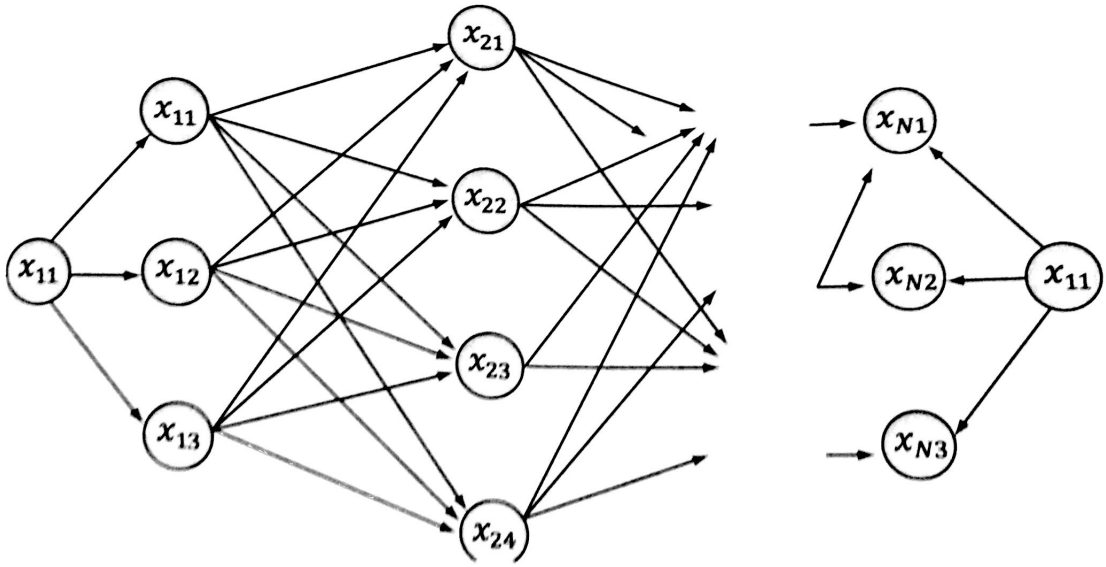
1. 维特比算法

   维特比算法是一个特殊但应用最广的动态规划算法。其是针对一个特殊图（篱笆网络，Lattice）的有向图最短路径问题提出的。凡事使用隐含马尔可夫模型描述的问题都可以用维特比算法来解码。

   假定输入的拼音是$y_1,y_2,\cdots,y_N$，对应的汉字是$x_1,x_2,\cdots,x_N$。

$$
\begin{aligned}
x_1, x_2, \cdots, x_N &= ArgMax_{x \in X} P(x_1, x_2, \cdots, x_N | y_1, y_2, \cdots, y_N) \\
&= ArgMax_{x \in X} \prod_{i=1}^{N} P(y_i | x_i) \cdot P(x_i | x_{i-1})
\end{aligned}
$$

相同的拼音可能对应不同的字，x_{ij}表示状态x_i的第j个可能的值。每个状态展开如下篱笆网络：



维特比算法的基础：

1. 如果概率最大的路径P（或最短路径）经过某个点，那么从起始点S到该点的子路径一定是起点到该点的最短路径。
2. 从S到E路径必经过第i时刻到某个状态。假定第i时刻有k个状态，如果记录了从S到第i个状态的所有k个节点的最短距离，最终最短路径必定经过其中一条。这样在任何时刻只需要考虑有限条候选路径。
3. 结合上述两点，假定从状态i进入状态i+1时，从S到状态i上各个节点的最短路径已找到，那么计算从S到第i+1状态的某节点最短路径时，只需考虑从S到状态i上k个节点的最短路径，以及这k个节点到$x_{i+1,j}$的距离即可。

维特比算法：

第一步，计算从S出发到第一个状态x_1的n_1个节点的距离$d(S, x_{1i})$. 因为是第一步，所以这些距离都是S到各自节点的最短距离。

第二步，对于第二个状态x_2的n_2个节点，计算出从S到它们的最短距离。
$d(S, x_{2i}) = min_{I=1,n_1}[d(S, x_{1j}) + d(x_{1j}, x_{2i})]$，时间复杂度为$O(n_1 \cdot n_2)$。

依次类推到最后一个状态。每一步计算的复杂度为相邻状态节点数目的乘积。假定节点最多的状态有D个节点（整个网络宽度为D），网格长度为N，整个维特比算法的复杂度为$O(N \cdot D^2)$。

2. CDMA（码分多址）技术——3G移动通信的基础

   扩频传输优点：抗干扰能力极强；信号难被截获；利用带宽更充分。

   频分多址（FDMA）时分多址（TDMA），CDMA对于频带和时间的利用率更高。

第27章：期望最大化算法

1. 文本的自收敛分类

   另外两种文本分类算法：

   - 利用实现设定好的类别对新文本分类（需要事先设定好类别和文本中心Centroids）
   - 自底而上将文本两两比较聚类（计算时间长）

   自收敛分类，随机挑选类别中心并优化，使其和真实的聚类中心尽可能一致（收敛）。

   分类步骤：

   1. 随机挑选K个点作为起始的中心。
   2. 计算所有点到这些聚类中心的距离，将这些点归到最近一类中。
   3. 重新计算每一类中心（取平均）。
   4. 重复2和3直到新中心和旧中心之间偏移很小，即过程收敛。

2. 期望最大化和收敛的必然性

   距离函数需要足够好，保证每次分类之后同一类各个点到中心的平均距离d较近，且不同类中心之间平均距离D较远。（d变小，D变大）

   期望最大化机器学习的一般思想：

   - 两个过程

     1. 根据现有的聚类结果对所有点重新进行划分。
     2. 根据重新划分结果，得到新的聚类。

   - 目标函数是点到聚类的距离d和聚类之间的距离D，整个过程是最大化目标函数。

   EM算法：

   - 期望值计算过程Expectation，根据现有模型计算各个观测数据输入到模型中的计算结果。
   - 最大化过程Maximization，重新计算模型参数，最大化期望值。

   当优化的目标函数是一个凸函数时，EM算法能保证获得全局最优解。（如熵函数，欧式距离）

第28章：逻辑回归和搜索广告

1. 搜索广告的发展

   第一阶段，按广告主出价高低来竞价排名广告。（早期Overture和百度）

   第二阶段，预测用户点击候选广告的概率（点击率预估）。

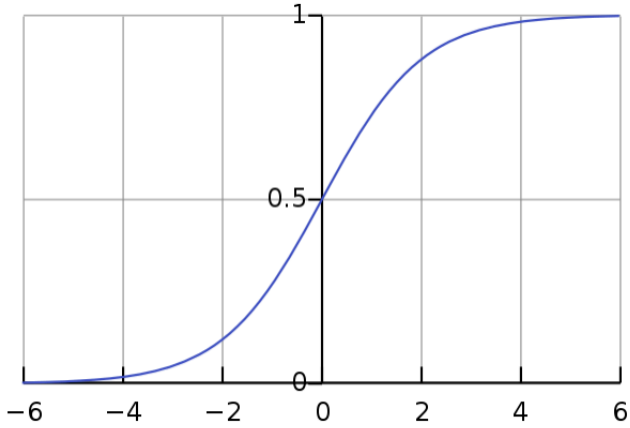
   第三阶段，进一步局部优化。

   多因素下统一数学模型预估点击率：逻辑回归模型（Logistic Regression/Logistic Model）

2. 逻辑回归模型

   逻辑回归模型指将一个事件出现的概率逐渐适应到一条逻辑曲线（Logistci Curve, 在(0, 1)之间，S型曲线，一开始变化快，逐渐变慢最后饱和）。如：

$$f(z)=\frac{e^z}{e^z+1}=\frac{1}{1+e^{-z}}$$



值域在$(0,1)$之间对应概率分布，自变量z的值在$(-\infty,+\infty)$之间。

假设有k个影响点击率的变量$x_1,x_2,\cdots,x_k$，线性组合：

$$z=\beta_0+\beta_1x_1+\beta_2x_2+\cdots+\beta_kx_k$$

x_i代表影响概率预测的各种信息，β_i代表相印变量的权重。具体问题中β可以由训练最大熵模型IIS方法得出。

第29章：各个击破算法和Google云计算的基础

1. 分治算法的原理

   云计算的关键之一是，如何把非常大的计算问题自动分解到许多计算能力有限的计算机上共同完成。Google的解决工具是MapReduce，其原理是常见的分治算法（Divide-and-Conquer）。

   分治算法：讲一个复杂问题分成若干简单的子问题进行解决。然后对子问题结果进行合并。

2. 从分治算法到MapReduce

   归并排序时间复杂度$O(Nlog_2N)$推导：

   假设N个元素的数组归并排序的算法计算时间为T(N)，则N/2个元素的子数组排序时间为T(N/2)，合并过程为N的线形函数。

$$\begin{aligned}T(N)&=2T(\frac{N}{2})+cN\\&=2[2T(\frac{N}{4})+\frac{cN}{2}]+cN\\&=2\{2(2T(\frac{N}{8})+\frac{cN}{4})+\frac{cN}{2}\}+cN\\&\cdots\\&=2^kT(1)+k\cdot cN\qquad(k=log_2N)\\&=N\cdot T(1)+cNlog_2N\\&=O(Nlog_2N)\end{aligned}$$

MapReduce的原理：将一台服务器无法存下的大数组计算拆分成多个部分到不同服务器中分别运算，然后合并。

假定矩阵有$A_{N\times N}$和$B_{N\times N}$，在10台服务器中计算$A\times B$。对矩阵A按行拆分，对矩阵B按列切分成10份。一台服务器完成十分之一的计算量，但每个元素的绝对计算时间相同。

若需要得到某个特定元素的值，可将矩阵A按列切分，B按行切分。每台服务器分别计算的结果相加得到该元素的值。

第30章Google大脑和人工神经网络

1. 人工神经网络（Artificial Neural Network）

   人工神经网络本质上是一种特殊的有向图：

   - 所有节点分层，每层节点可以通过有向弧指向上层节点且没有同层或跃层有向弧。
   - 每条弧上有一个权重（权值）。

   神经网络的语音识别应用：实际声学模型一般以元音辅音为单位建立，每个元音或辅音对应的一组数据可以看成多维空间中的一个区域。识别这些语音实际是在多维空间中划分一些区域。

   在人工神经网络中需要设计的是：

   - 结构，如层数，节点数，节点连接方式等
   - 非线性函数的设计，常用指数函数$f(G)=e^{w_0+w_1x_1+w_2x_2+\cdots+w_nx_n}$。它的模式分类能力等价于最大熵模型。

2. 训练人工神经网络

   弧上权重通过训练得到。人工神经网络的训练方式分为有监督的训练（Supervised Training）和无监督训练（Unsupervised Training）。

   有监督：

   事先有标注好的样本（既有输入数据也有对应的输出值）。定义成本函数fc（Cost Function）表示神经网络的到的输出值和实际值之间的差距。训练目标是找到参数$\hat{w}$使得：

   $$\hat{w}=AegMin_w fc$$

   通常用梯度下降法（Gradient Descent）来寻找最优。

   无监督：

   事先的样本中只有输入数据没有输出。需要根据实际应用设计合适的成本函数，如多维空间模式分类可以把每一个样本点到训练出来的聚类中心的欧几里得距离的均值作为成本函数；语言模型的条件概率可以用熵作为成本函数。定义成本函数之后用梯度下降法进行无监督的参数训练。

3. 人工神经网络与贝叶斯网络的关系

   共同点：

   - 都是有向图。每一节点取值只取决于前一级的节点，而与更前面的节点无关。
   - 训练方法相似
   - 对于很多模式分类问题效果（准确性）相似。
   - 计算量特别大

不同点：

- 人工神经网络结构上完全标准化，贝叶斯网络更灵活。
- 神经网络中变量只能先进行线性组合最后对一个变量进行非线性变换。贝叶斯网络中变量可以组成任意函数。灵活度增加且复杂度也增加。
- 贝叶斯网络更容易考虑前后相关性，可以解码一个输入序列。神经网络很难处理序列，主要应用估计一个概率模型的参数。

4. Google大脑

   Google大脑是一种大规模并行处理的人工神经网络。

   Google大脑采用神经网络而不是其他机器学习技术的原因：

   - 人工神经网络可以在多维空间画出各种形状的模式分类边界，有很好的通用性。
   - 算法非常稳定。
   - 易于并行实现。

第31章：大数据的威力

1. 数据的重要性

   广义的数据包括信息和情报。人类的文明与进步，某种意义上是通过对数据进行收集、处理和总结而成的。

   在没有数据之前不要给出任何结论。

2. 数据的统计和信息技术

   统计学是通过搜索、整理、分析数据等手段，以达到推断所测对象的本质，甚至预测对象未来的一门综合性科学。

   统计首先要求数据量充足。

   切比雪夫不等式：

$$P(|X-E(X)|\geq\epsilon)<\frac{\sigma^2}{n\epsilon^2}$$

   其中X是一个随机变量，E(X)是该变量的数学期望，n是试验次数（样本数），ϵ是误差，σ是方差。这个公式的含义为，当样本数足够多时，一个随机变量和它数学期望之间的误差可以任意小。

   统计还要求采样的数据具有代表性。

3. 为什么需要大数据

   大数据数据量大，多维度和完备性，可以将看似无关的事件联系起来，恢复对事物全方位完整的描述。