

# 信息检索与数据挖掘

---

## 第9章 基于语言建模的检索模型

# 课程内容

- 第1章 绪论
- 第2章 布尔检索及倒排索引
- 第3章 词项词典和倒排记录表
- 第4章 索引构建和索引压缩
- 第5章 向量模型及检索系统
- 第6章 检索的评价
- 第7章 相关反馈和查询扩展
- 第8章 概率模型
- **第9章 基于语言建模的检索模型**
- 第10章 文本分类
- 第11章 文本聚类
- 第12章 Web搜索
- 第13章 多媒体信息检索
- 第14章 其他应用简介

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

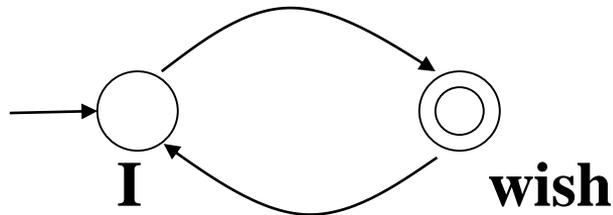
- 语言模型如何应用到**IR** 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

# 最简单的语言生成模型

## generative model

一个简单的**有穷自动机**及其生成语言中的一些字符串。  
→ 指向的是自动机的**初始状态**，而双圈节点对应的是  
(可能的) **终止状态**



**I wish**

**I wish I wish**

**I wish I wish I wish**

**I wish I wish I wish I wish**

...

**\*wish I wish**

如果上述自动机是带有概率的，  
则是概率语言模型(probabilistic LM)

也称统计语言模型(Statistical Language Modeling, SLM)

# 有穷自动机 → 语言模型

- 一个语言模型 (language model) 是从某词汇表上抽取的**字符串到概率的一个映射函数**。也就是说，对于字母表 $\Sigma$ 上的语言模型M有  $\sum_{s \in \Sigma^*} P(s) = 1$
- 最简单的语言模型仅包含一个节点，只有一个生成不同词项的概率分布，因此有  $\sum_{t \in V} P(t) = 1$

## Model M

		<b>the</b>	<b>man</b>	<b>likes</b>	<b>the</b>	<b>woman</b>
<b>0.2</b>	<b>the</b>	——	——	——	——	——
<b>0.1</b>	<b>a</b>					
<b>0.01</b>	<b>man</b>	<b>0.2</b>	<b>0.01</b>	<b>0.02</b>	<b>0.2</b>	<b>0.01</b>
<b>0.01</b>	<b>woman</b>					
<b>0.03</b>	<b>said</b>					
<b>0.02</b>	<b>likes</b>					
<b>...</b>						

一个具体的字符串或文档的概率往往非常小

multiply

$P(s | M) = 0.00000008$

# 统计语言模型(Statistical Language Modeling)

## 一个概率语言模型 (SLM) 的例子

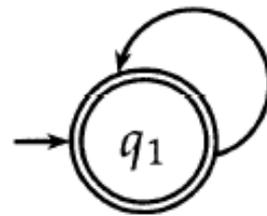
- 单状态概率有穷状态自动机——一元语言模型——状态发射概率分布如右表。其中**STOP**不是词，而是表示自动机结束的一个标识符。这样，概率

$$\begin{aligned}
 P(\text{frog said that toad likes frog}) &= (0.01 \times 0.03 \times 0.04 \times 0.01 \times 0.02 \times 0.01) \\
 &\quad \times (0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.2) \\
 &\approx 0.000\ 000\ 000\ 001\ 573
 \end{aligned}$$

the	0.2
a	0.1
frog	0.01
toad	0.01
said	0.03
likes	0.02
that	0.04
...	...

第一行是词项发射概率；第二行是生成每个词后继续前进或停止的概率

一个有穷自动机要转变成为一个良构 (well-formed) 的语言模型，必须要有一个显式的**停止概率**



$$P(\text{STOP}|q_1) = 0.2$$

# 比较两个语言模型 $M_1$ 和 $M_2$

这里我们是对**概率求积**，但是通常在概率应用中，实际上往往采用**对数求和**的计算方法

若  $P(d|M_1) > P(d|M_2)$   
说明d由 $M_1$ 生成的可能性大

模式 $M_1$		模式 $M_2$	
the	0.2	the	0.15
a	0.1	a	0.12
frog	0.01	frog	0.0002
toad	0.01	toad	0.0001
said	0.03	said	0.03
likes	0.02	likes	0.04
that	0.04	that	0.04
dog	0.005	dog	0.01
cat	0.003	cat	0.015
monkey	0.001	monkey	0.002
...	...	...	...

图 12-3 两个一元语言模型的部分概率赋值

$s$	frog	said	that	toad	likes	that	dog
$M_1$	0.01	0.03	0.04	0.01	0.02	0.04	0.005
$M_2$	0.0002	0.03	0.04	0.0001	0.04	0.04	0.01

$$P(s|M_1) = 0.000\ 000\ 000\ 000\ 48 ,$$

$$P(s|M_2) = 0.000\ 000\ 000\ 000\ 000\ 384$$

$$P(s | M_1) > P(s | M_2)$$

# 语言模型的比较

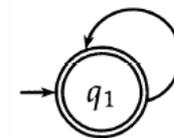
- 比较两个模型，可计算**似然比**（**likelihood ratio**），即将其中一个模型的数据生成概率除以另外一个模型的数据生成概率。

$$O(R | \bar{x}, \bar{q}) = \frac{P(R = 1 | \bar{x}, \bar{q})}{P(R = 0 | \bar{x}, \bar{q})}$$

- 假定**停止概率**是固定的，那么考虑它并不会改变似然比的顺序，也不会改变两个语言模型生成同一字符串的概率大小次序。因此，它不会影响文档的排序。
- 课程中后续内容将**忽略停止概率**

# 小结：语言模型

$$\begin{aligned}
 P(\text{frog said that toad likes frog}) &= (0.01 \times 0.03 \times 0.04 \times 0.01 \times 0.02 \times 0.01) \\
 &\quad \times (0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.2) \\
 &\approx 0.000\ 000\ 000\ 001\ 573
 \end{aligned}$$



$$P(\text{STOP}|q_1) = 0.2$$

the	0.2
a	0.1
frog	0.01
toad	0.01
said	0.03
likes	0.02
that	0.04
...	...

- 比较两个模型，可计算**似然比**（likelihood ratio），我们忽略**停止概率**
- 若语言模型与文档是一一映射的关系，那么**查询与文档的相关性**可以转化为**查询与语言模型的相关性**

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到IR 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

# 对于词项序列如何求解其生成的概率值？

- 根据链式规则将一系列事件的概率分解成多个连续的事件概率之积，每个概率是每个事件基于其历史事件的条件概率。具体计算公式如下：

$$P(t_1t_2t_3t_4) = P(t_1)P(t_2/t_1)P(t_3/t_1t_2)P(t_4/t_1t_2t_3)$$

- 最简单的语言模型形式是：去掉所有条件概率中的条件来独立地估计每个词项的概率。这种模型称为**一元语言模型**（unigram language model）：

$$P_{uni}(t_1t_2t_3t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

词袋模型 Bag of words

# 由一元语言模型产生一个文档d的概率？ 词的多项式分布

- 一元语言模型会给有序的词项序列赋予概率。当然，这些词项按照其他次序出现的概率也等于这个概率。因此，实际上这相当于**词项存在一个多项式分布**。我们也可以将上述模型称为多项式模型。

$$P(d) = \frac{L_d!}{tf_{t_1,d}! tf_{t_2,d}! \cdots tf_{t_M,d}!} P(t_1)^{tf_{t_1,d}} P(t_2)^{tf_{t_2,d}} \cdots P(t_M)^{tf_{t_M,d}}$$

$L_d = \sum_{1 \leq i \leq M} tf_{t_i,d}$  是文档的长度（即词条的总个数）

M是词项词典的大小

$$P(X_1 = x_1, \dots, X_k = x_k) = \begin{cases} \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k} & \text{when } \sum_{i=1}^k x_i = n \\ 0 & \text{otherwise.} \end{cases}$$

多项式分布的概率公式

# 怎样由文档生成语言模型？ $\leftarrow M_D$ 的估计

- 问题：已知样本 $D$ ，求其模型 $M_D$ 的参数 $P(w/M_D)$ 。
- 对于该参数估计问题，可以采用最大似然估计 (**Maximum Likelihood Estimation, MLE**)。
- **MLE**：使得观察样本出现概率(似然)最大的估计。
  - 一射击世界冠军和一菜鸟打靶，其中一人放一枪得到10环，请问是谁打的？显然世界冠军打的可能性大，也就是说这是使得10环这个事件出现概率最大的估计。

# $M_D$ 的MLE估计

- 设词项词典大小为 $L$ ，则模型 $M_D$ 的参数可以记为：

$$\begin{aligned}\vec{\theta}_D &= (\theta_1, \theta_2, \dots, \theta_L) \\ &= (P(w_1 | M_D), P(w_2 | M_D), \dots, P(w_L | M_D))\end{aligned}$$

- **MLE估计：**

$$\vec{\theta}_D^* = \arg \max_{\vec{\theta}_D} P(D | \vec{\theta}_D)$$

- 关键是如何求  $P(D | \vec{\theta}_D)$ ，也就是说假设这些参数未知的情况下，如何求上述概率。

# 文本生成的多项式模型

- 有一个 $L$ 个面的不规则骰子，在第 $i$ 个面上写着 $w_i$ ，文档 $D=d_1d_2\dots d_n$ 可以认为是抛 $n$ 次骰子得到的
- 检索过程就是根据观察样本 $D$ 的估计 $Q$ 的生成概率，即在已知抛 $n$ 次的结果为文档 $D$ 的条件下，抛 $m$ 次的结果为查询 $Q$ 的概率 $P(Q|M_D)=?$
- $D = (c(w_1,D),c(w_2,D),\dots,c(w_L,D))$ ， $c(w_i,D)$ 是文档 $D$ 中 $w_i$ 的出现次数
- $D =$  我喜欢基于统计语言模型的信息检索模型
- $D = (<我,1>,<喜欢,1>,<基于,1>,<统计,1>,<语言,1>,<模型,2>,<的,1>,<信息,1>,<检索,1>)$

# $M_D$ 的参数求解

• 求解

$$\vec{\theta}_D^* = \arg \max_{\vec{\theta}_D} P(D | \vec{\theta}_D) = \arg \max_{\vec{\theta}_D} n! \prod_{i=1}^L \frac{\theta_i^{c(w_i, D)}}{c(w_i, D)!}$$

$$\sum_{i=1}^L \theta_i = 1$$

多项式分布的概率公式

$$P(X_1 = x_1, \dots, X_k = x_k) = \begin{cases} \frac{n!}{x_1! \dots x_k!} p_1^{x_1} \dots p_k^{x_k} & \text{when } \sum_{i=1}^k x_i = n \\ 0 & \text{otherwise.} \end{cases}$$

• 条件极值问题，采用拉格朗日法求解，得到拉格朗日函数：

$$L(\lambda, \vec{\theta}_D) = n! \prod_{i=1}^L \frac{\theta_i^{c(w_i, D)}}{c(w_i, D)!} + \lambda(1 - \sum_{i=1}^L \theta_i)$$

• 对每个 $\theta_i$ 求偏导，令其为0，解得：

$$\theta_i^* = P_{ML}(w_i | M_D) = \frac{c(w_i, D)}{\sum_{j=1}^L c(w_j, D)} = \frac{c(w_i, D)}{|D|}$$

# 一个MLE估计的例子

- $D = (\langle \text{我}, 1 \rangle, \langle \text{喜欢}, 1 \rangle, \langle \text{基于}, 1 \rangle, \langle \text{统计}, 1 \rangle, \langle \text{语言}, 1 \rangle, \langle \text{模型}, 2 \rangle, \langle \text{的}, 1 \rangle, \langle \text{信息}, 1 \rangle, \langle \text{检索}, 1 \rangle)$

- 采用MLE估计有：

$$\begin{aligned} P(\text{我}|M_D) &= P(\text{喜欢}|M_D) = P(\text{基于}|M_D) \\ &= P(\text{统计}|M_D) = P(\text{语言}|M_D) = P(\text{的}|M_D) \\ &= P(\text{信息}|M_D) = P(\text{检索}|M_D) = 0.1 \end{aligned}$$

$$P(\text{模型}|M_D) = 0.2$$

其他词项的概率为0

# 小结：怎样由文档生成语言模型？

- 一元语言模型（unigram language model）：

$$P_{uni}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

- 问题：已知样本 $D$ ，求其模型 $M_D$ 的参数 $P(w/M_D)$ 。

$$P(d) = \frac{L_d!}{tf_{t_1,d}! tf_{t_2,d}! \cdots tf_{t_M,d}!} P(t_1)^{tf_{t_1,d}} P(t_2)^{tf_{t_2,d}} \cdots P(t_M)^{tf_{t_M,d}}$$

$$\vec{\theta}_D = (\theta_1, \theta_2, \dots, \theta_L)$$

$$= (P(w_1 | M_D), P(w_2 | M_D), \dots, P(w_L | M_D))$$

$M_D$ 的参数求解

$$\vec{\theta}_D^* = \arg \max_{\vec{\theta}_D} P(D | \vec{\theta}_D)$$

$$\theta_i^* = P_{ML}(w_i | M_D) = \frac{c(w_i, D)}{\sum_{j=1}^L c(w_j, D)} = \frac{c(w_i, D)}{|D|}$$

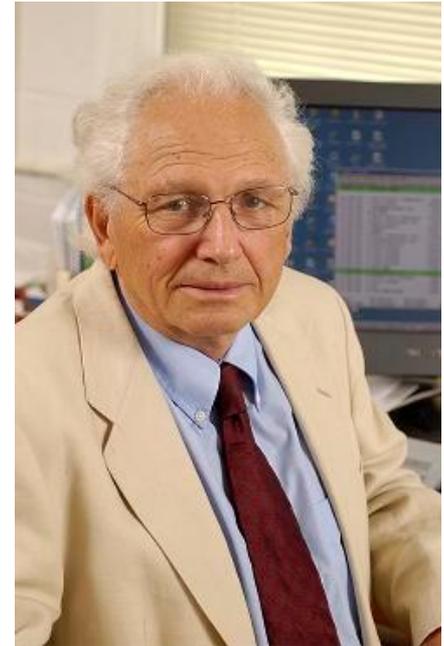
# 自然语言处理大师

## Frederick Jelinek

Frederick Jelinek (1932 – 2010) was a Czech-American researcher in information theory, automatic speech recognition, and natural language processing. He is well known for his oft-quoted statement, "Every time I fire a linguist, the performance of the speech recognizer goes up".

[https://en.wikipedia.org/wiki/Frederick\\_Jelinek](https://en.wikipedia.org/wiki/Frederick_Jelinek)

<http://www.clsp.jhu.edu/~jelinek/>



弗莱德里克·贾里尼克(Fred Jelinek)从麻省理工获得博士学位后，在哈佛大学教了一年书，然后到康乃尔大学任教。他之所以选择康乃尔大学，是因为找工作时和那里的一位语言学家谈得颇为投机……贾里尼克在康乃尔十年磨一剑，潜心研究信息论，终于悟出了自然语言处理的真谛。1972年，贾里尼克到IBM华生实验室（IBM T.G.Watson Labs）做学术休假，无意中领导了语音识别实验室，两年后他在康乃尔和IBM之间选择了留在IBM。在那里，贾里尼克组建了阵容空前绝后强大的研究队伍……

——吴军《数学之美》

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到IR 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

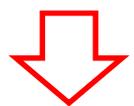
# 语言模型的种类：n-gram

- 一元语言模型（unigram language model），也称上下文无关语言模型



$$P_{uni}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

- 二元语言模型（bigram language model），即计算条件概率时只考虑前一个词项的出现情况：



$$P_{bi}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2/t_1)P(t_3/t_2)P(t_4/t_3) \quad \text{一阶马尔科夫链}$$

- 三元语言模型（trigram language model）
- ...
- **Unigram → bigram → ... → n-gram**

还有其他更复杂的种类，本课程不介绍

# 类比：打扑克中的出牌策略



一元模型：只根据当前牌出牌；

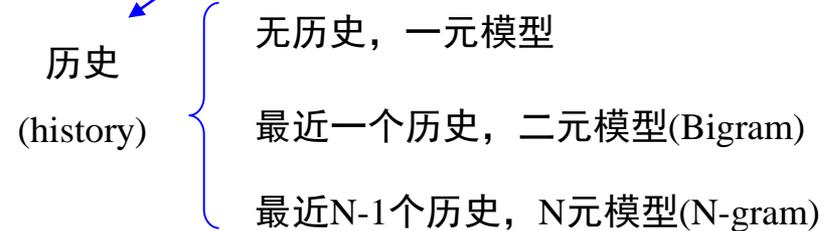
二元模型：根据上一轮牌出牌；

.....

# 语言模型复杂度

- 对于一个文档片段  $d = w_1 w_2 \cdots w_n$ ，统计语言模型是指概率  $P(w_1 w_2 \cdots w_n)$  求解，由链式法则有

$$P(w_1 w_2 \cdots w_n) = P(w_1) P(w_2 \cdots w_n | w_1) = \dots = P(w_1) \prod_{i=2}^n P(w_i | w_{i-1} \cdots w_1)$$



对于n-gram，**n越大，则模型越复杂**，估计的参数(即估计的概率)也越多。当然，当数据量足够大的情况下，模型阶数越高对片段概率的计算也越准确。

**将统计语言模型用于信息检索领域的大部分工作都只使用了一元模型**

# 利用N-gram模型生成文档的复杂度

- **1-gram**模型需要顾及的参数个数为 $|V|$

$$\vec{\theta}_D = (\theta_1, \theta_2, \dots, \theta_L)$$

$$= (P(w_1 | M_D), P(w_2 | M_D), \dots, P(w_L | M_D))$$

- **N-gram**模型中，需要计算系统词表中词的每一个N-1元组的概率，共有 $|V|^{(N-1)}$ 个元组，有 $|V|^{(N-1)}$ 个不同的概率分布。对于每一个分布，又必须估算N个参数，共需估算出 $|V|^N$ 个参数。如 $|V|=10000$ ， $N=3$ ，就必须计算出 $10^{12}$ 个参数。由于参数空间随着N的增大呈指数上升，因此，N一般取2或3。
- **N-gram**模型的优点在于它包含了前N-1个词所能提供的全部信息，这些信息对当前词出现具有很强的约束力。缺点在于需要相当规模的训练文本来确定模型的参数。当N较大时，模型的参数空间过大。

估计的参数数目为： $|V| + |V|^2 + \dots + |V|^N = (|V|^{N+1} - |V|) / (|V| - 1)$

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到IR 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

# 语言模型如何应用到IR 中？

## • IR中使用LM的问题

- N个文档，各自有一个语言模型，给定一个查询，求查询与哪个文档相关度最高？

## • 对比问题

- 问题：设有N个作者，每人有一篇文章，对于不在上述N篇文章中的一篇新文档Q，问最有可能是哪个作者写的？
- 一个解决思路：根据每个作者写的文章，总结出作者的写作风格，然后根据写作风格来判断Q与谁的风格最近。

**一种可能的思路：把相关度看成是每篇文档对应的语言模型下生成该查询的可能性**

# 总体分布&抽样

- 文档的模型(风格)实际上是某种**总体分布**
- 文档和查询都是该总体分布下的一个**抽样样本实例**
- 根据文档，估计文档的模型，即求出该**总体分布**(一般假设某种总体分布，然后求出其**参数**)
- 然后**计算该总体分布下抽样出查询的概率**

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到IR 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

## IR 中查询似然模型(Query Likelihood Model)

- IR 中最早使用也是最基本的语言模型是查询似然模型 (query likelihood model)。在这个模型中，我们对文档集中的每篇文档 $d$  构建其对应的语言模型 $M_d$ 。我们的目标是将文档按照其与查询相关的似然 $P(d/q)$  排序。
- 由贝叶斯公式： $P(d/q) = P(q/d)P(d)/P(q)$
- $P(q)$ 对所有的文档都一样，因此可以被忽略。文档的先验概率 $P(d)$ 往往可以视为均匀分布，因此也可以被省略。
- 实现目标的途径：最后我们会按照 $P(q/d)$ 进行排序，它是在文档 $d$  对应的语言模型下生成 $q$  的概率。因此，IR 中的语言建模方法实际上是在对查询的生成过程进行建模：首先每篇文档对应一个文档模型，然后计算查询被视为每个文档模型的随机抽样样本的概率，最后根据这些概率对文档排序。

# 理论上计算 $P(d/q)$ 的方法

用 $M_d$ 替代 $d$

- 最普遍的计算 $P(d/q)$ 的方法是使用一元语言模型

$$P(q | M_d) = K_q \prod_{t \in V} P(t | M_d)^{tf_{t,d}}$$

$K_q = L_d^{-1} (tf_{t_1,d}! tf_{t_2,d}! \cdots tf_{t_M,d}!)$

- $K_q$ 是查询 $q$ 的多项式系数，对于某个特定的查询，它是一个常数，因此可以被忽略。在基于语言模型（简记为LM）的检索中，可以将查询的生成看成一个随机过程。具体的方法是：
  - (1) 对每篇文档推导出其LM
  - (2) 估计查询在每个文档 $d_i$ 的LM下的生成概率 $P(q | M_{d_i})$
  - (3) 按照上述概率对文档进行排序

## 实际中，如何估计概率 $P(q | M_{d_i})$ ?

- 在MLE（maximum likelihood estimation，最大似然估计）及一元语言模型假设的情况下，给定文档d的LM  $M_d$  的情况下生成查询q的概率为

$$\hat{P}(q | M_d) = \prod_{t \in q} \hat{P}_{\text{mle}}(t | M_d) = \prod_{t \in q} \frac{tf_{t,d}}{L_d}$$

- 其中， $M_d$  是文档d的LM， $tf_{t,d}$  是词项t在文档d中出现的原始频率， $L_d$  是文档d中的词条数目。  
即，我们只是将词在文档中出现的次数除以文档中出现的所有的词的总数。

## 小结：查询似然模型

- **查询似然模型（query likelihood model, QLM）**

- 目标：将文档按照其与查询相关的似然 $P(d|q)$  排序
- 实现目标的途径：按照 $P(q|d)$ 进行排序

- **具体的方法是：**

- (1) 对每篇文档推导出其LM
- (2) 估计查询在每个文档 $d_i$  的LM 下的生成概率 $P(q | M_{d_i})$

$$P(q | M_d) = K_q \prod_{t \in V} P(t | M_d)^{tf_{t,d}}$$


$$\hat{P}(q | M_d) = \prod_{t \in q} \hat{P}_{\text{mle}}(t | M_d) = \prod_{t \in q} \frac{tf_{t,d}}{L_d}$$

- (3) 按照上述概率对文档进行排序

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到**IR** 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

# 估计 $P(q | M_{d_i})$ ，零概率是个问题

- 将不出现在文档中的词项  $t$  的概率  $\hat{P}(t | M_d)$  估计成 **0**?

$$\hat{P}(q | M_d) = \prod_{t \in q} \hat{P}_{\text{mle}}(t | M_d) = \prod_{t \in q} \frac{tf_{t,d}}{L_d}$$

表中  $df_t$  是包含  $t$  的文档数目

- 与  $RSV_d$  的估计类似

	文档	相关	不相关	总计
词项出现	$x_t=1$	$s$	$df_t-s$	$df_t$
词项不出现	$x_t=0$	$S-s$	$(N-df_t)-(S-s)$	$N-df_t$
	总计	$S$	$N-S$	$N$

$c_t$  查询词项的优势率比率 (odds ratio) 的对数值

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{1-p_t} + \log \frac{1-u_t}{u_t}$$



$$p_t = s/S, \quad u_t = (df_t-s)/(N-S)$$

$$c_t = K(N, df_t, S, s) = \log \frac{s / (S - s)}{(df_t - s) / ((N - df_t) - (S - s))}$$

# 平滑

## 其他的平滑方法?

- 在减少出现事件的概率估计值的同时提高未出现事件的概率估计值的方法称为平滑 (smoothing)

能出现的0 概率?

$$c_t = K(N, df_t, S, s) = \log \frac{s / (S - s)}{(df_t - s) / ((N - df_t) - (S - s))}$$

一种最简单的平滑方法就是对每个观察到的事件的数目都加上一个数 $\alpha$  相当于在所有词汇表上使用了均匀分布作为一个贝叶斯先验  $\alpha$ 的大小表示我们对均匀分布的信心强度

$$\hat{c}_t = K(N, df_t, S, s) = \log \frac{(s + \frac{1}{2}) / (S - s + \frac{1}{2})}{(df_t - s + \frac{1}{2}) / (N - df_t - S + s + \frac{1}{2})}$$

## 其他平滑的方法

### 线性插值LM (linear interpolation LM)

- 如果  $\mathbf{tf}_{t,d} = 0$ ，那么有  $\hat{P}(t|M_d) \leq \mathbf{cf}_t / T$
- 其中， $\mathbf{cf}_t$  是  $t$  在整个文档集的出现次数， $T$  是所有文档集中词条的个数。一个实际效果较好的简单方法是，将基于**文档**的多项式分布和基于全部**文档集**估计出的多项式分布相**混合**，即

$$\hat{P}(t | d) = \lambda \hat{P}_{\text{mle}}(t | M_d) + (1 - \lambda) \hat{P}_{\text{mle}}(t | M_c)$$

- 其中， $0 < \lambda < 1$ ，**Mc**是基于全部文档集构造的LM。上述公式混合了来自单个文档的概率词在整个文档集的出现频率。该模型中**如何设置正确的 $\lambda$** 是获得良好性能的关键。

# 线性插值LM示例

- 在语言建模的IR模型下，查询 $q$ 的检索排序函数定义如下

$$P(d | q) \propto P(d) \prod_{t \in q} [(1 - \lambda)P(t | M_c) + \lambda P(t | M_d)]$$

- 例12-3 假定某文档集中包含如下两篇文档：
  - d1: Xyzy reports a profit but revenue is down
  - d2: Quorus narrows quarter loss but revenue decreases further
- 混合参数 $\lambda = 1/2$ 。假定查询为revenue down，则有

$$\begin{aligned} P(q|d_1) &= [(1/8 + 2/16)/2] \times [(1/8 + 1/16)/2] \\ &= 1/8 \times 3/32 = 3/256, \end{aligned}$$

?

$$\begin{aligned} P(q|d_2) &= [(1/8 + 2/16)/2] \times [(0/8 + 1/16)/2] \\ &= 1/8 \times 1/32 = 1/256, \end{aligned}$$

## 另外一种方法

- 另外一种方法是，将从全部文档集中获得的语言模型（不是看成均匀分布）看成贝叶斯更新过程（**Bayesian updating process**）的一个先验分布。于是有

$$\hat{P}(t | d) = \frac{tf_{t,d} + \alpha \hat{P}(t | M_c)}{L_d + \alpha}$$

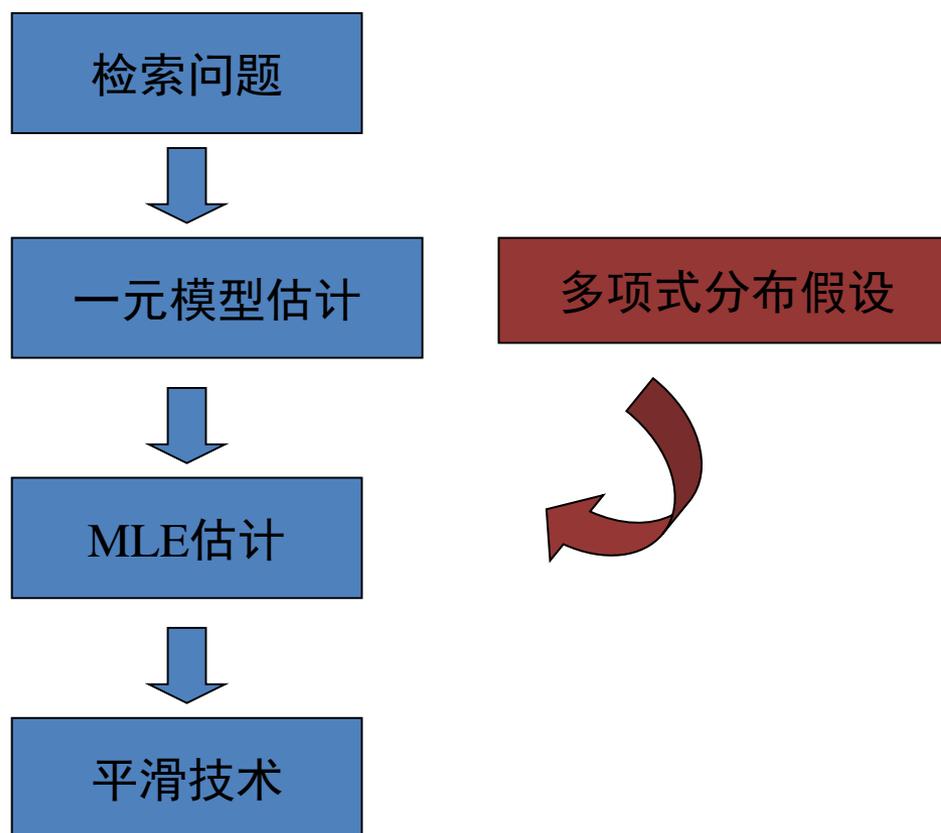
- 两种方法中，对文档中出现的词，它的概率估计是将一个折损的MLE估计和它在整个文档集中的流行度因子相结合而得到的。而对于文档中没有出现的词，则直接采用词在整个文档集中的流行度因子来计算。

# 语言模型的应用效果

- **Ponte 和 Croft (1998)** 首次将语言建模的方法引入到IR领域并进行了实验。表明语言建模方法得到的词项权重计算方法优于传统的tf-idf方法。

Rec.	tf-idf	precision		
		LM	%chg	
0.0	0.7439	0.7590	+2.0	
0.1	0.4521	0.4910	+8.6	
0.2	0.3514	0.4045	+15.1	*
0.3	0.2761	0.3342	+21.0	*
0.4	0.2093	0.2572	+22.9	*
0.5	0.1558	0.2061	+32.3	*
0.6	0.1024	0.1405	+37.1	*
0.7	0.0451	0.0760	+68.7	*
0.8	0.0160	0.0432	+169.6	*
0.9	0.0033	0.0063	+89.3	
1.0	0.0028	0.0050	+76.9	
Ave	0.1868	0.2233	+19.55	*

# 小结：QLM(Query Likelihood Model)模型



# 本讲内容：基于语言建模的检索模型

- 语言模型

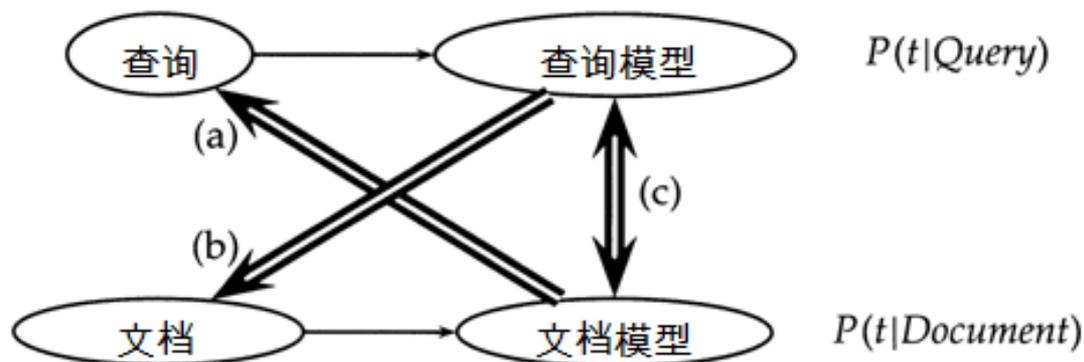
- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到**IR** 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法

# 扩展的LM方法

- 查询似然类：文档建模，计算查询的似然，例子--基本QLM模型、**翻译模型**等
- 文档似然类：查询建模，计算文档的似然，例子--**BIM模型**、相关性模型(Relevance模型)等
- 模型比较类：文档建模、查询建模，计算两个模型的距离，**KL距离模型**



IR 中使用统计语言建模的 3 种方式：(a) 查询似然；(b) 文档似然；(c) 模型比较

# KL距离模型

$$D(P \parallel Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)}$$

- 通过计算查询模型和文档模型的KL距离（Kullback-Leibler divergence）来对返回文档d的风险进行建模

$$R(d; q) = LK(M_d \parallel M_q) = \sum_{t \in V} P(t | M_q) \log \frac{P(t | M_q)}{P(t | M_d)}$$

- KL 距离是源自信息论的一个非对称距离度量方法，主要度量的是概率分布 $M_q$ 对 $M_q$ 建模的无效程度（参见Cover和Thomas 1991及Manning和Schütze 1999）。Lafferty和Zhai（2001）给出的结果表明基于模型对比的方法比查询似然和文档似然的方法都好。
- 使用KL 距离作为排序函数的一个缺点是最后的得分在查询之间没有可比性。这对于ad hoc 检索来说没有关系，但是对于一些其他应用（如话题跟踪）却影响很大。

# 翻译模型

- 基本的LM 方法没有考虑表达方式不同的问题，比如一义多词或者查询语言和文档语言间的偏差问题。翻译模型可以通过翻译的方法，让不在文档中的词项转变为其近义词项后出现在查询中。这种方法也为跨语言IR 提供了基础。假定翻译模型可以通过一个词项间的条件概率分布  $T(\cdot|\cdot)$  来表示，则基于翻译的查询生成模型可以定义为：

$$P(q | M_d) = \prod_{t \in q} \sum_{v \in V} P(v | M_d) T(t | v)$$

- 其中， $P(v|M_d)$ 是基本的文档语言模型， $T(t|v)$ 表示的是翻译概率。

# 本讲内容：基于语言建模的检索模型

- 语言模型

- 什么是概率语言模型？如何比较两个模型？
- 怎样由文档生成语言模型？
- 语言模型的种类

- 语言模型如何应用到**IR** 中？

- 查询似然模型(Query Likelihood Model)
- 平滑的方法：线性插值LM
- 扩展的LM 方法