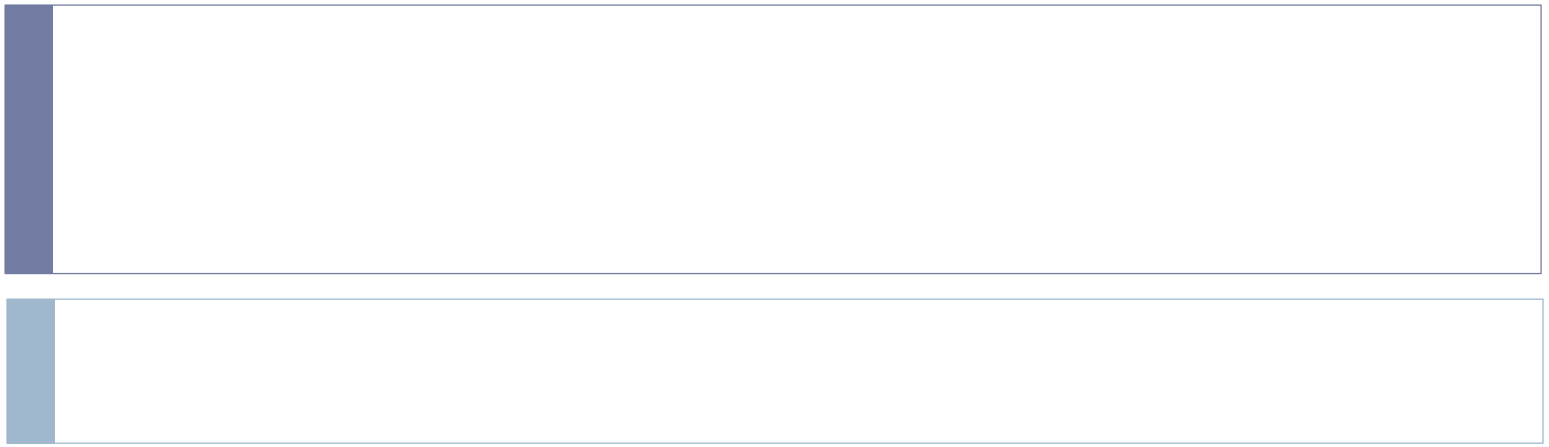
第2章 信息检索模型

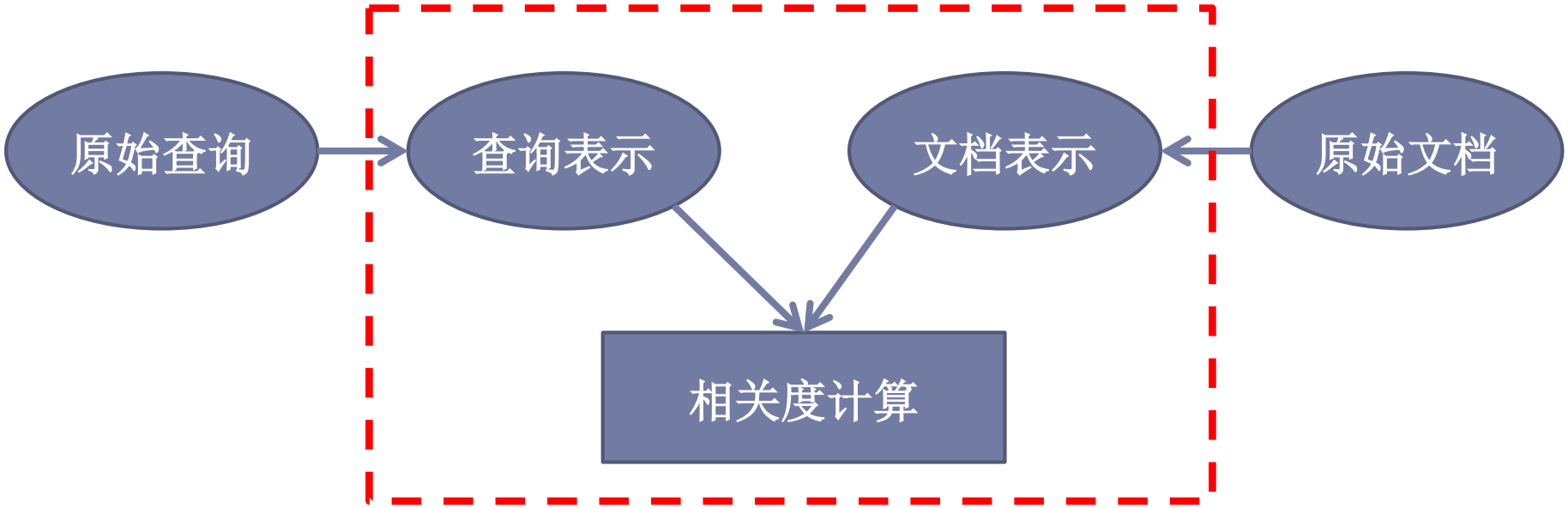


Index

- 模型定义及分类
- 布尔模型
- 向量空间模型
- 隐性语义索引模型(LSI: Latent Semantic Indexing)
- 基于概率统计的模型
  - BM25

IR Model

- 信息检索模型是IR中的核心内容之一。
- 信息检索模型是指如何对查询和文档进行表示，然后对它们进行相似度计算的框架和方法。
  - 本质上是对相关度建模。



基本概念

- 索引项(Index Term)
  - 展现文档主题的基本单位
  - 文档表示成多个Term的集合
  - 通常用词来表示，但是也可以用其他语言单位来表示
  - 关键词(key words) 可以看成Term的一种
  - 不同索引项描述文档内容的能力不同
- 索引项的权重(Weight)
  - 刻画索引项描述文档语义内容的重要性
  - 通常是非负实数
  - 通过权重区分索引项的重要性

IR Model的三个关键问题

- 标引项Term的选择
- 权重计算(Term Weighting)
- 查询和文档的相似度计算(Similarity Computation)

文本的表示

- 词袋 (bag of words, BOW)
  - 以词或词组(或n-gram)的语义单元作为特征项(Term)
  - 在特征项的频率基础上进行文本的表示、特征处理和统计学习
  - 忽略语义单元间的联系及词序
- "John is quicker than Mary"以及"Mary is quicker than John"这两句话对应相同的表示

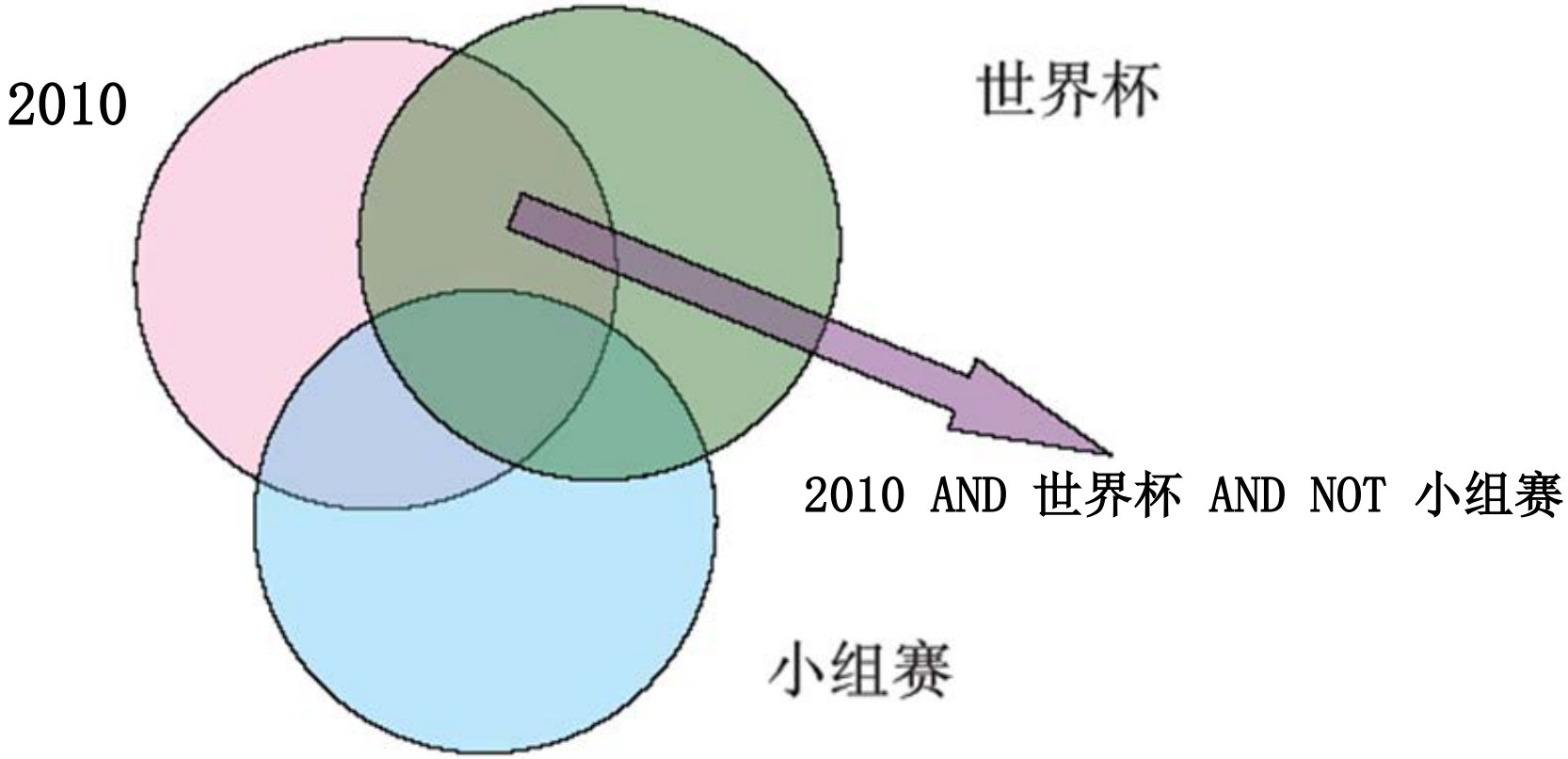
检索模型分类

- 根据使用的数学方法分类
  - 基于集合论的模型(Set Theoretic models)
    - 布尔模型
    - 扩展布尔模型
    - 基于模糊集的模型
  - 基于代数论的模型(Algebraic models)
    - 向量空间模型
    - 隐性语义索引模型
  - 基于概率统计的模型(Probabilistic models)
    - 回归模型
    - 二元独立概率模型
    - 语言模型

布尔模型

- 查询和文档均表示为Term(“是否存在”)的布尔表达式
  - 所有Term(存在)的“与”关系
  - 例子：
    - 查询：2010 AND 世界杯 AND NOT 小组赛
    - 文档1：2010年世界杯在南非举行。
    - 文档2：2010年世界杯小组赛已经结束。
- 相似度计算
  - 查询和所有文档的布尔表达式进行匹配，匹配成功的文档得分为1，否则为0。
  - 类似于传统数据库检索，是精确匹配

布尔模型匹配的集合表示



布尔模型的优缺点

- 优点：
  - 简单：现代很多搜索引擎中仍然包含布尔模型的思想，如Google的高级检索
  - 自我保护功能：暗暗地降低用户对搜索系统的期望，使自己不在责任方，检索结果不好的原因在于用户构造查询不好
- 缺点：
  - 只能严格匹配(得分不是0就是1)，不能近似或者部分匹配，多个结果无法排序
  - 一般用户构造查询不是很容易，构造不利可能造成结果过多或者过少

向量

- 向量(矢量，vector)：既有大小又有方向的量，通常用有向线段表示，记作$\vec{x}$
- 考虑从空间坐标系原点出发(其他向量可以平移到原点出发)的向量$\vec{x}$，终点坐标为$<x_1,x_2,\ldots,x_n>$，称之为一个n维向量
- 向量的运算：加、减、倍数、内积

$$\vec{x} \pm \vec{y} =< x_1 \pm y_1, x_2 \pm y_2, \ldots, x_n \pm y_n >$$

$$\lambda \vec{x} =< \lambda x_1, \lambda x_2, \ldots, \lambda x_n >$$

$$\vec{x} \bullet \vec{y} =< x_1 \times y_1, x_2 \times y_2, \ldots, x_n \times y_n >$$

向量的模、距离和夹角

- 向量的模(大小):

$$\left|\vec{x}\right| = \left\|\vec{x}\right\| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$$

- 向量的(欧氏)距离

$$dist(\vec{x}, \vec{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}$$

- 夹角余弦

$$\cos\alpha = \frac{\vec{x} \bullet \vec{y}}{\left|\vec{x}\right| \times \left|\vec{y}\right|}$$

向量空间模型(Vector Space Model，VSM)

- Cornell大学Salton等人上世纪70年代提出并倡导，原型系统SMART
- Term独立性假设
  - Term在文档中的出现是独立、互不影响的。
- 基本思想
  - 一篇文档(查询或待查询文档)表示为特征空间中的一个向量，称为文档向量(也可以将其视为空间中的点)。
  - 文档向量中每一维对应于文档中的一个特征项，它的权值为该向量维对应的特征在文档库中的权值。
  - 通过计算向量间的距离即可得到查询和每个文档的相似度。

向量空间模型

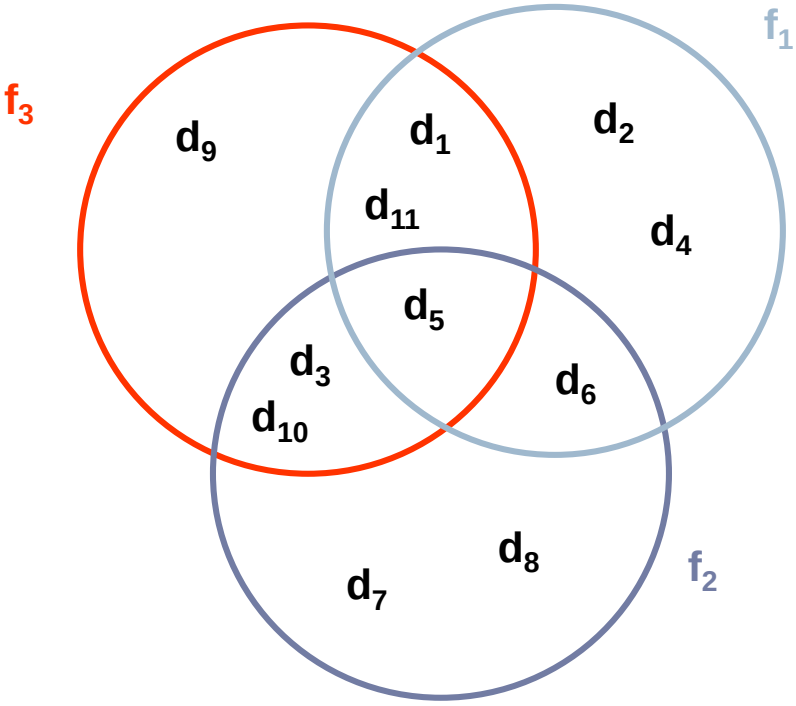
- 文档←→向量
  - 根据标引词表(包含N个标引词)，建立|N|-维向量空间
  - 标引词是空间中的轴
  - 文档是空间中的点或向量
  - 向量空间特征
    - 高维：应用于Web搜索引擎时，可能产生高达百万维的向量空间
    - 稀疏：很多项都为0
- 查询←→向量
  - 表示方法同上
- 查询与文档的相似度～向量相似度
  - 查询与文档的相似度与向量间距离成反比

(1)Term的选择

- Term是能代表文档内容的特征
- Term粒度
  - Term可以是字、词、短语、N-gram或者某种语义单元，最简单的是采用全文标引(full text indexing)，即用文档中出现的所有的字或者词作为标引词。
- 降维
  - VSM中向量的维数很大(以中文词索引为例，向量维数会上10万)时，往往也同时引入了很多噪音。
  - 如：去停用词、对英文进行词干还原、只选择名词作为Term、将Term聚成的各个不同类作为一个Term、选择出现次数较多的词作为Term等

文档的VSM表示

文档	*f1*	*f2*	*f3*
d1	**1**	**0**	**1**
d2	**1**	**0**	**0**
d3	**0**	**1**	**1**
d4	**1**	**0**	**0**
d5	**1**	**1**	**1**
d6	**1**	**1**	**0**
d7	**0**	**1**	**0**
d8	**0**	**1**	**0**
d9	**0**	**0**	**1**
d10	**0**	**1**	**1**
d11	**1**	**0**	**1**



简单组合

(2)权重计算

- 布尔权重：term i在文档 j 中的权重a_{ij}=0 or 1(出现则取1，否则取0)

$$w_{ij} = \begin{cases} 1 & f_i \in d_j \\ 0 & f_i \notin d_j \end{cases}$$

文档	*f1*	*f2*	*f3*
d1	**1**	**0**	**1**
d2	**1**	**0**	**0**
d3	**0**	**1**	**1**
d4	**1**	**0**	**0**
d5	**1**	**1**	**1**
d6	**1**	**1**	**0**
d7	**0**	**1**	**0**
d8	**0**	**1**	**0**
d9	**0**	**0**	**1**
d10	**0**	**1**	**1**
d11	**1**	**0**	**1**

(2)权重计算

- 影响权值的因素
  - Intra-cluster similarity
    - 某索引项在文档中出现的次数越多，则该索引项越能表示文档的内容
    - TF
  - Inter-cluster similarity
    - 含有某索引项的文档数越少，则表明该索引项在区分相似文档的能力越强
    - IDF
  - 两者之间的平衡
    - TF*IDF

(2)权重计算

- TF(Term Frequency)：term在文档中出现的次数。权重$a_{ij}=TF_{ij}$或者归一化后的TF值
  - TF的归一化(Normalization)：将一篇文档中所有Term的TF值归一化到[0,1]之间。通常可以采用以下三种方式之一：
    - Maximum Normalization
      [1,2,1,0,4]->[0.25,0.5,0.25,0,1]

$$\frac{TF_i}{\underset{i}{Max}\, TF_i}$$

    - Augmented Maximum Normalization
      [1,2,1,0,4]->[0.625,0.75,0.625,0.5,1]

$$0.5+0.5\times\frac{TF_i}{\underset{i}{Max}\, TF_i}$$

    - Cosine Normalization
      [1,2,1,0,4]->[0.213,0.426,0.213,0,0.852]

$$\frac{TF_i}{\sqrt{\sum_i TF_i^2}}$$

TF Maximum Normalization举例

文档	*f1*	*f2*	*f3*
d1	**2**	**0**	**3**
d2	**4**	**0**	**0**
d3	**0**	**3**	**6**
d4	**5**	**0**	**0**
d5	**3**	**2**	**5**
d6	**5**	**5**	**0**

文档	*f1*	*f2*	*f3*
d1	**0.66**	**0**	**1**
d2	**1**	**0**	**0**
d3	**0**	**0.5**	**1**
d4	**1**	**0**	**0**
d5	**0.6**	**0.4**	**1**
d6	**1**	**1**	**0**

(2)权重计算

- IDF(Inverse Document Frequency)
  - term的文档频率DF(Document Frequency)：term在整个文档集合中出现的文档篇数。DF反映了term的区分度，DF越高表示term越普遍，区分度越低，因此权重也越低。
  - 逆文档频率(Inverse DF，IDF)：DF的倒数，通常采用如下公式进行计算(N是文档集合中所有文档的数目)：

$$IDF = \frac{N}{DF}$$

IDF举例

- 很少在文档集中出现的索引项，其IDF值较高
- 经常在文档集中出现的索引项，其IDF值较低

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{1000}\right) = 1$$

$$\log\left(\frac{10000}{10}\right) = 3$$

$$\log\left(\frac{10000}{1}\right) = 4$$

(2)综合权重计算

- VSM中通常采用TF*IDF的方式计算权重
  - term i在文档d_j中的权重$a_{ij}=TF_{ij}*IDF_i$
- 对TF或IDF的缓冲
  - 对TF进行缓冲：1+log(TF), 1+log(1+log(TF))
  - 对IDF进行缓冲：1+log(N/DF)
    - log的作用：将值域拉平，使得函数的变化更平缓(常常以e为底，即ln)
    - 加1的作用：平滑

$$w_{ij} = \frac{TF(f_i, d_j)}{\max_k TF(f_K, d_j)} \times \left(\log \frac{m}{DF(f_j)} \right)$$

$$w_{t,d} = (1 + \log tf_{t,d}) \times \log_{10} N / df_t$$

(2)权重计算中存在的问题

- 文档长度因素
  - 文档长度大小不一
  - 关于文档长度的两个观点：
    - 长文档具有更多的词
    - 长文档具有更多的信息
- 常常需要对长文档进行惩罚，对短文档进行补偿。
  - Pivoted Normalization：

    1- b+b*doc_len/avg_doc_len (b在0~1之间)

(3)相似性度量

- 相似性函数
  - 计算两个向量之间的相似程度
  - 根据相似性值大小对检索的结果进行排序
  - 通过给定阈值，限定检索返回结果的文档数量

相似性函数

- 距离函数：假定用n个特征来描述向量，那么就可以把每个向量看作是n维空间中的一点，进而可以使用某种距离来表示向量之间的相似性，距离较近的向量较相似，距离较远的向量则差异比较大。由此可见**距离函数**刻画的是样本之间的**不相似性**。

- 常见距离函数
  - 绝对值距离 $$D(V_1,V_2)=\sum_{i=1}^{m}\left|w_{1i}-w_{2i}\right|$$
  - 欧式距离 $$D(V_1,V_2)=\sqrt{\sum_{i=1}^{m}\left(w_{1i}-w_{2i}\right)^2}$$
  - 闵氏距离 $$D(V_1,V_2)=\sqrt[q]{\sum_{i=1}^{m}\left(w_{1i}-w_{2i}\right)^q},q>0$$

距离—相似性变换

- 距离函数代表向量间的不相似程度，直接用来处理有关向量相似性问题不方便。
- 相似系数：相似系数基于如下假定，两个向量愈相似，则相似系数愈大；向量愈不相似，则相似系数愈小。
- 通常利用数学变换，使得距离函数可以直接用来表示相似性

$$S(V_1,V_2)=e^{-D(V_1,V_2)}$$

三维空间

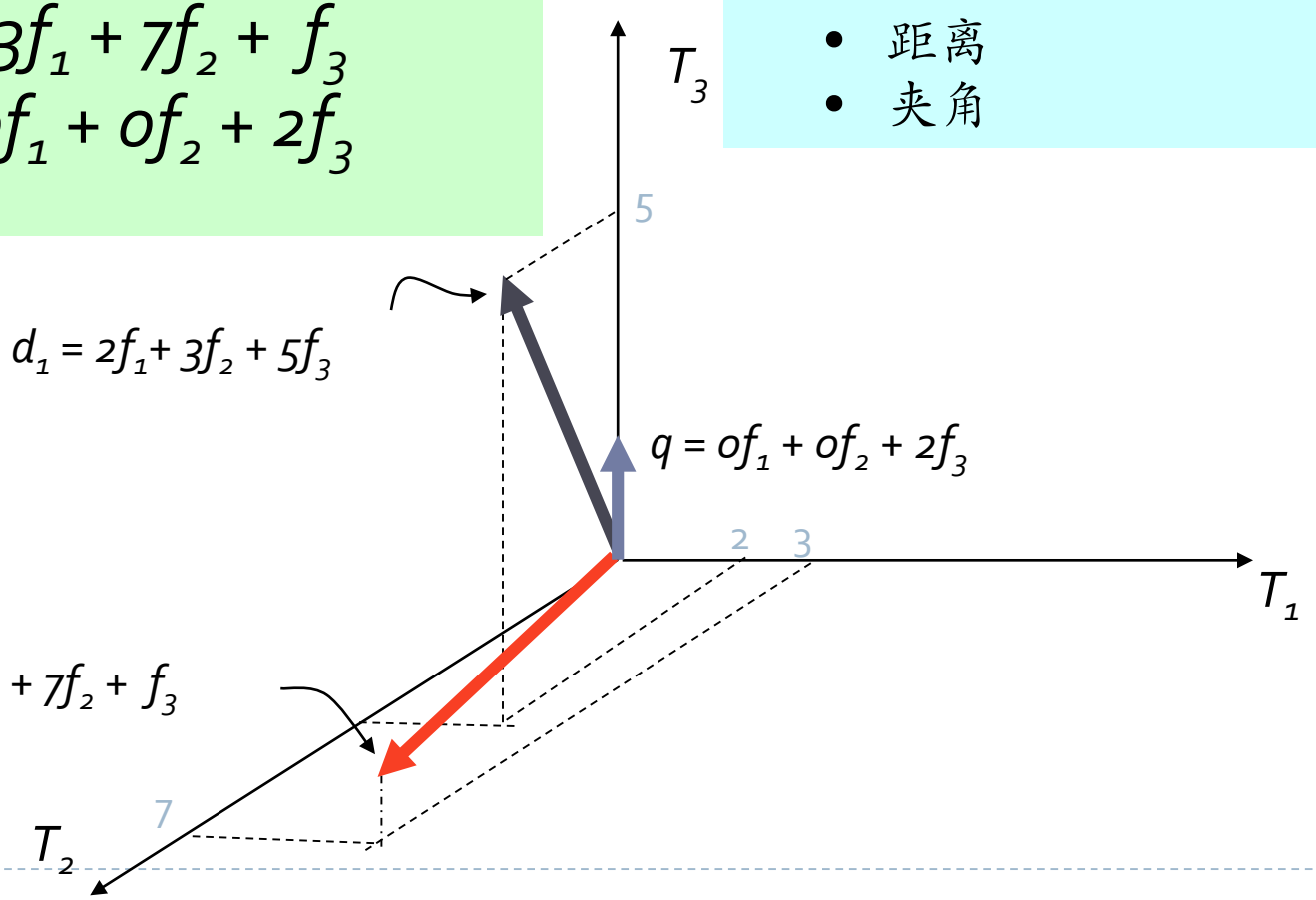
Example:

$d_1 = 2f_1 + 3f_2 + 5f_3$

$d_2 = 3f_1 + 7f_2 + f_3$

$q = 0f_1 + 0f_2 + 2f_3$

- d_1 和 d_2 谁与 q 更相似?
- 相似性度量?
  - 距离
  - 夹角



距离还是夹角？

- 通过实验验证
  - 将一篇文档d的内容完整复制两次，放在同一篇文档内，生成新的文档d′。
  - 从语义角度而言，文档d和d′具有相同的内容
- 使用Euclidean距离衡量文档相似度时，上述两个向量间的距离非常大
- 使用角度衡量文档相似度时，两个文档间的夹角为0，也就是具有最大相似度

长度归一化

- 向量除以其自身的模，归一化因子称为L_2-norm

$$\|\vec{x}\|_2 = \sqrt{\sum_i x_i^2}$$

- 利用L_2-norm，可以使参与计算的向量具有相同的单位长度
  - 通过归一化，d 与d′ 可表示为相同的向量

角度->夹角余弦

- 利用文档与查询向量间的夹角计算，完成对文档的排序
  - 查询向量与文档向量间的夹角越大，相似度越低
  - 夹角余弦Cosine 在 [0°, 180°] 区间内单调递减
  - 查询向量与文档向量的夹角余弦值越大，相似度越高，余弦值越小，相似度越低

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\bullet\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\bullet\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

  - q_i 和 d_i 分别是索引项 i 在查询、文档向量中的tf-idf权重

例子

- 查询q：(<2010,1>,<世界杯,2>)
- 文档d_1：(<2010,1>,<世界杯,3>,<南非,1>,<举行,1>)
- 文档d_2：(<2002,1>,<世界杯,2>,<韩国,1>,<日本,1>,<举行,1>)

	d_1	d_2	q
2002	0	1	0
2010	1	0	1
世界杯	3	2	2
南非	1	0	0
韩国	0	1	0
日本	0	1	0
举行	1	1	0

例子

- 查询和文档向量的相似度计算：
  - 距离：
    - 文档d_1与q的欧氏距离：

$$\sqrt{(1-1)^2+(3-2)^2+1^2+1^2}=\sqrt{3}$$

    - 文档d_2与q的欧氏距离：

$$\sqrt{1^2+(0-1)^2+(2-2)^2+1^2+1^2+1^2}=\sqrt{5}$$

  - 夹角余弦：
    - 文档d_1与q的夹角余弦：$\frac{7}{\sqrt{12\times 5}}\approx 0.90$
    - 文档d_2与q的夹角余弦：$\frac{4}{\sqrt{5\times 8}}\approx 0.63$

相似系数

- Simple matching Coefficient $|Q \cap D|$
- Dice's Coefficient $2\frac{|Q \cap D|}{|Q|+|D|}$
- Jaccard's Coefficient $\frac{|Q \cap D|}{|Q \cup D|}$
- Cosine Coefficient $\frac{|Q \cap D|}{|Q|^{1/2} \times |D|^{1/2}}$
- Overlap Coefficient $\frac{|Q \cap D|}{\min(|Q|,|D|)}$

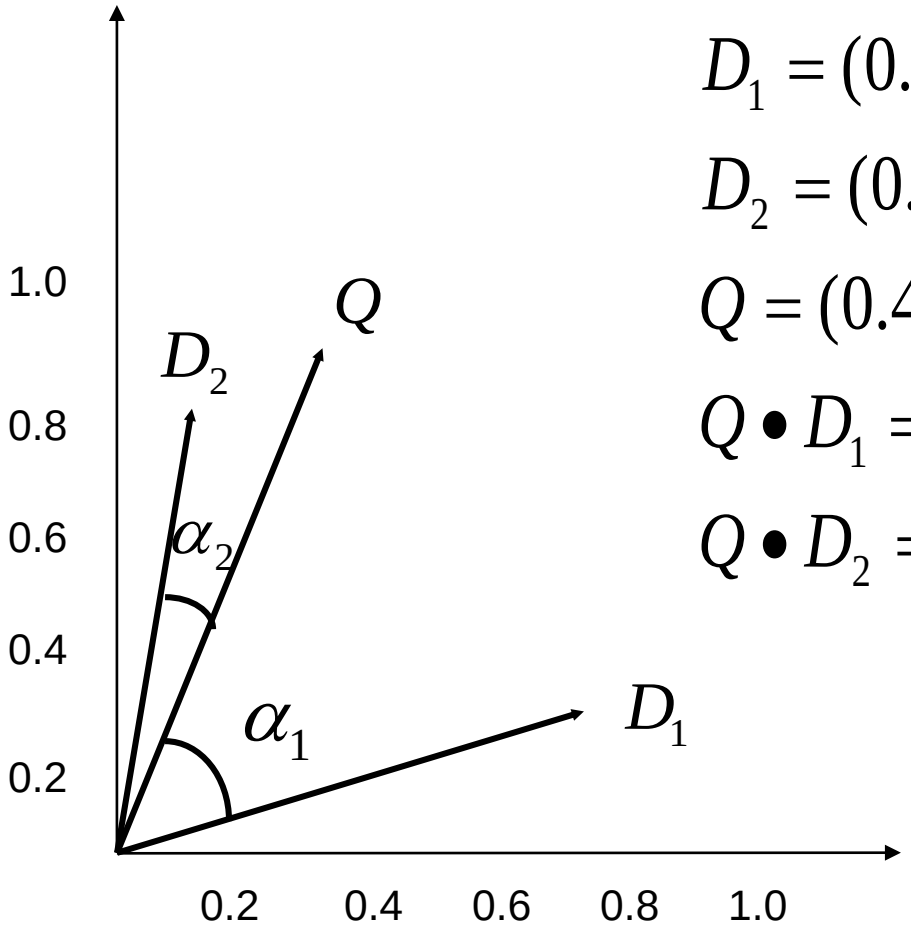
Simple matching coefficient

- 向量内积 (Inner Product)
  - 查询与文档的相似度由其向量内积来表示：

$$sim(q, d_j) = q \bullet d_j = \sum_{i=1}^{n} w_{iq} \times w_{ij}$$

  - 对于布尔权值，内积就是同时出现在文档和查询的索引项数目

Simple matching coefficient



$$D_1 = (0.8, 0.3)$$

$$D_2 = (0.2, 0.7)$$

$$Q = (0.4, 0.8)$$

$$Q \bullet D_1 = 0.8 \times 0.4 + 0.3 \times 0.8 = 0.56$$

$$Q \bullet D_2 = 0.2 \times 0.4 + 0.7 \times 0.8 = 0.64$$

VSM的优缺点

- 优点：
  - 简洁直观，可以应用到很多其他领域(文本分类、生物信息学)
  - 支持部分匹配和近似匹配，结果可以排序
  - 检索效果不错
- 缺点：
  - 理论不足：基于直觉的经验性公式
  - 标引项之间的独立性假设与实际不符：实际上，Term的出现之间是有关系的，不是完全独立的。如：“Jobs”和“Apple”的出现不是独立的。

隐性语义索引(LSI)

- 向量空间模型中向量表示中的其他问题：
  - 一词多义Polysemy: Java (language/island)
  - 一义多词Synonymy: 电脑　计算机
  - 词并不等价于概念或者语义
- LSI(Latent Semantic Indexing)
  - Susan Dumais等人提出*
  - 利用数学中的SVD分解(奇异值分解)，重新表示索引项
  - 去掉噪音
  - LSA: Latent Semantic Analysis

*Dumais, S. T., Furnas, G. W., Landauer, T. K. and Deerwester, S. (1988), "Using latent semantic analysis to improve information retrieval." In Proceedings of CHI'88: Conference on Human Factors in Computing, New York: ACM, 281-285.

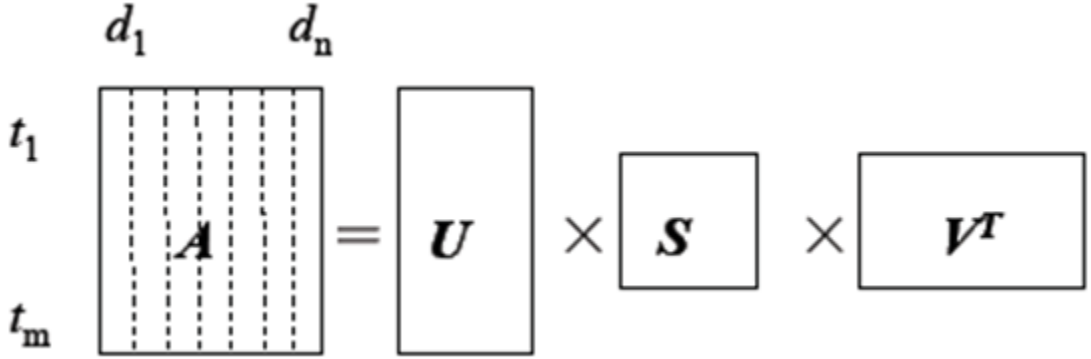
VSM扩展——LSI

- 文档-索引项矩阵(Doc-Term Matrix)
  - n篇文档，m个索引项构成的矩阵A_{m*n}
  - 每列是每篇文档的向量表示
  - 每行是索引项的向量表示
  - A^TA表示文档两两之间的相似度

$$A_{m*n} = \begin{array}{c} \\ t_1 \\ t_2 \\ \vdots \\ t_m \end{array} \begin{array}{c} \begin{array}{cccc} d_1 & d_2 & \dots & d_n \end{array} \\ \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \end{array}$$

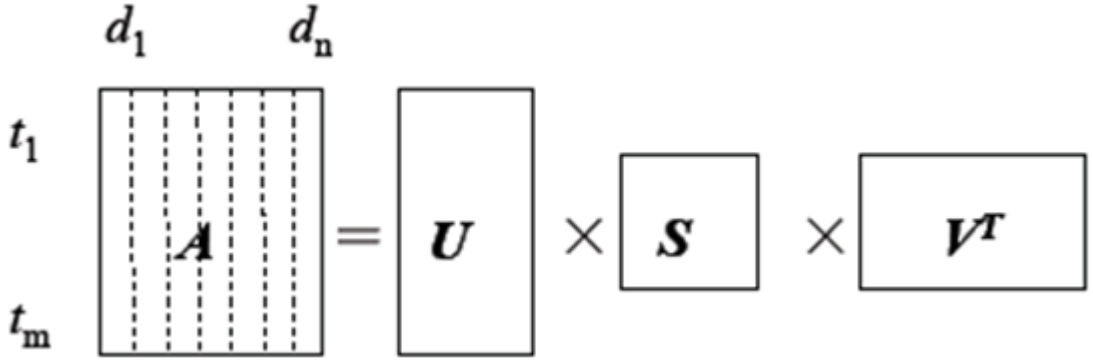
奇异值分解(Singular Value Decomposition，SVD)

- $A_{m*n}=USV^T$



  - U为m*r矩阵
  - S为r*r对角且从左上到右下降序排列的方阵，对角线上的值称为Singular Value
  - V为n*r矩阵
  - r是A的秩(A中不为零的子式最高阶数r)

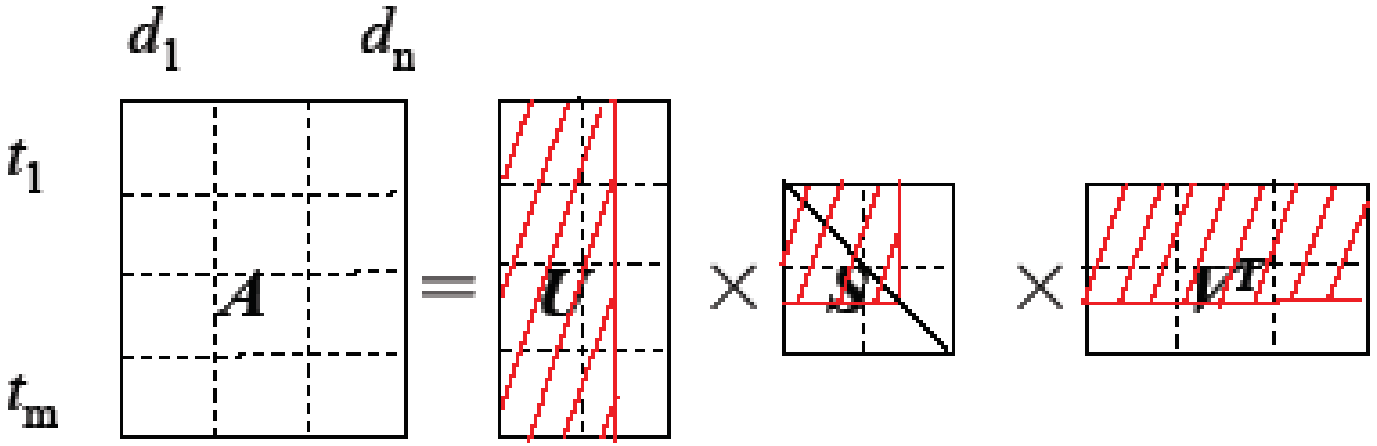
矩阵的物理意义



- 矩阵U中的每一行表示意思相关的一类词，其中的每个非零元素表示这类词中每个词的重要性（或者说相关性），数值越大越相关。
- V^T中的每一列表示同一主题一类文档，其中每个元素表示这类文档中每篇文档的相关性。
- S矩阵则表示类词和文档类之间的相关性。
- 只要对关联矩阵A进行一次奇异值分解，就同时完成了近义词分类和文档的分类。

去噪

- S中从左上角到右下角只选择较大的k个Singular value，得到方阵S'。同样U取前k列，V取前k列分别得到U'和V'，令A'=U'S'V'^{T}，A'是A的近似
- 新矩阵A'是A的一个k-秩近似矩阵，它在最小平方意义下最接近原始矩阵，即最优的近似矩阵。



新矩阵A’

- A’包含了A的主要结构信息，可以理解为对A的重构，它忽略了词项使用上的噪音数据，由于维数的降低，近似的词项被合并。
- 同义词在k维空间中有相似的表示，并且在这个k维空间中，出现在相似文档中的词项也将是近似的，即使它们从未出现在同一个文档中。
- LSI构造了新的语义空间，具备“概念检索”的特征。
- 降维因子K的选取非常关键，一方面，K应该足够大，以反映原始数据的信息与结构；另一方面，K应该足够小，以便过滤掉所有不相关的冗余信息及细节(或噪音)。

实例

Index Words	Titles								
	T1	T2	T3	T4	T5	T6	T7	T8	T9
book			1	1					
dads						1			1
dummies		1						1	
estate							1		1
guide	1					1			
investing	1	1	1	1	1	1	1	1	1
market	1		1						
real							1		1
rich						2			1
stock	1		1					1	
value				1	1				

book	0.15	-0.27	0.04
dads	0.24	0.38	-0.09
dummies	0.13	-0.17	0.07
estate	0.18	0.19	0.45
guide	0.22	0.09	-0.46
investing	0.74	-0.21	0.21
market	0.18	-0.30	-0.28
real	0.18	0.19	0.45
rich	0.36	0.59	-0.34
stock	0.25	-0.42	-0.28
value	0.12	-0.14	0.23

*

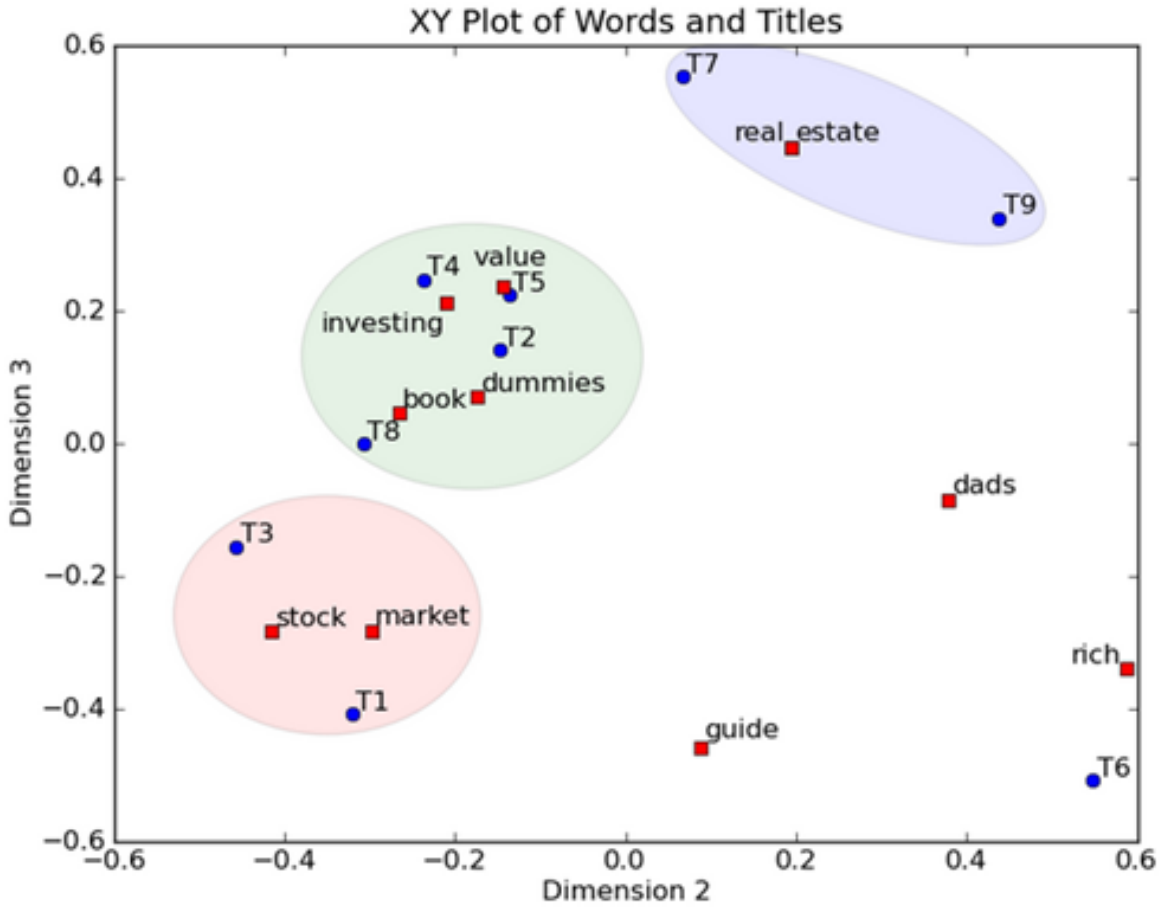
3.91	0	0
0	2.61	0
0	0	2.00

*

T1	T2	T3	T4	T5	T6	T7	T8	T9
0.35	0.22	0.34	0.26	0.22	0.49	0.28	0.29	0.44
-0.32	-0.15	-0.46	-0.24	-0.14	0.55	0.07	-0.31	0.44
-0.41	0.14	-0.16	0.25	0.22	-0.51	0.55	0.00	0.34

实例

- 取k=2，投影到一个平面上



基于LSI的查询及更新策略

- 将查询q看成一个虚拟文本向量放入，经过LSI得到A’，将q向量和其他文档向量进行向量计算，则可以得到该查询和所有文档的相似度。
- SVD更新策略
  - 对已经进行了奇异值分解的词频矩阵，若有新的文档或词项加入，主要有两种方法进行SVD更新：重新计算SVD或者直接加入。
  - 直接加入是一种简单的更新策略，如下图分别为直接加入p个文档或q个词项

SVD更新策略

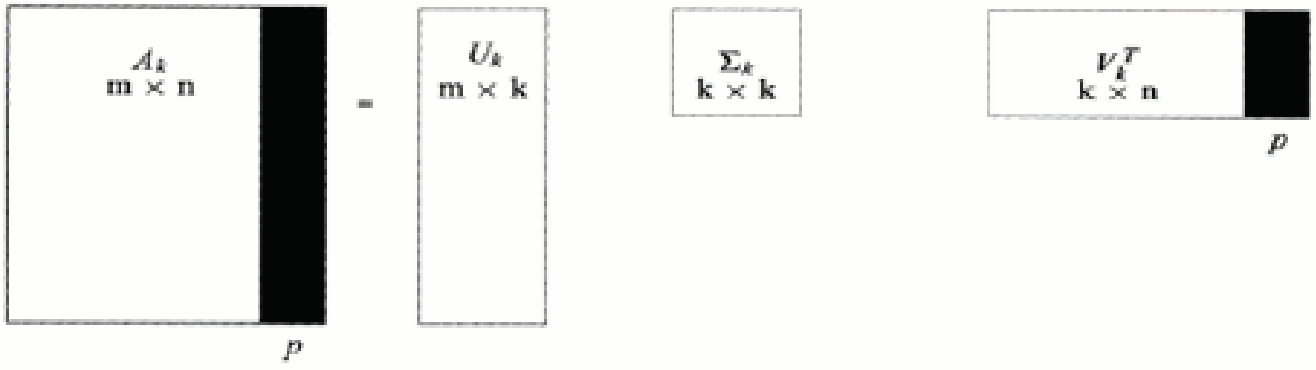


FIG. 2. *Mathematical representation of folding-in p documents.*

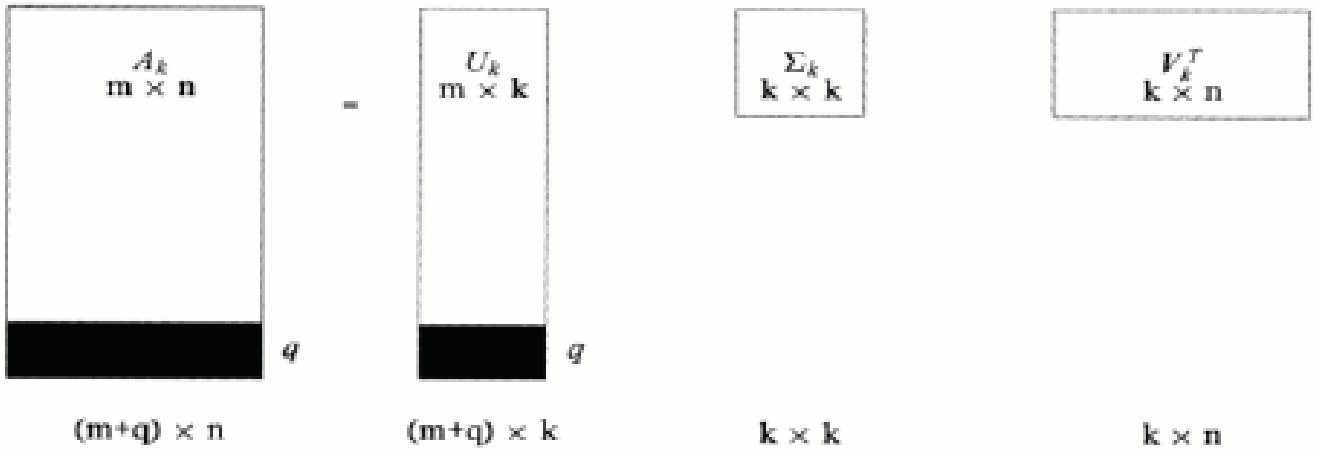


FIG. 3. *Mathematical representation of folding-in q terms.*

有关LSI的实验结论

- SVD非常耗时，目前还没有特别快的方法
- k通常取经验值(200~1000)
- 在一些小规模语料(如CACM)上面效果不错，但是在大规模语料(如TREC中的语料)上效果一般。
- 可以应用到分类、过滤、跨语言检索等多个领域。

(3)概率模型

- 概率排序原则
- 基本思想
  - 给定用户查询q和文档集合中文档d_j的概率模型，估计用户查询q与文档d_j相关的概率$P(R=1|d_j,q)$，再对结果进行排序
- Okapi BM25
  - 基于词项频率、文档长度等因子来建立概率模型

Introduction to BM25

- BMXX
  - Best match
  - Ranking/Retrieval function
  - Probabilistic Relevance Framework (PRF)
  - Rank matching documents according to their relevance to a given search query
- Okapi BM25
  - first system to implement BM25 function
  - BM0,BM1,BM11,BM15

BM functions

(BM0) $$w = 1$$

(BM1) $$w = \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)}$$

(BM15) $$w = \frac{tf}{(k_1 + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} + k_2 \times nq \frac{(\Delta - d)}{(\Delta + d)}$$

(BM11) $$w = \frac{tf}{(\frac{k_1 \times d}{\Delta} + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} + k_2 \times nq \frac{(\Delta - d)}{(\Delta + d)}$$

Okapi BM25

- BM25(
- score
  follows

qtf

$c(q,Q)$:
k_3 : par

k_1 : parameter
b: [0,2), slope parameter
$C(q,D)$: raw TF of q in D
$|D|$: document length
$avdl$: average document length

Inverse Doducment Frequency

$$\sum_{q \in Q \cap D} \frac{(k_3 + 1$$
$$k_3 +$$

$$dtf(q,D) = \frac{(k_1+1)\cdot c(q,D)}{k_1\left(1-b+b\frac{|D|}{avdl}\right)+c(q,D)} = \frac{(k_1+1)\cdot c'(q,D)}{k_1+c'(q,D)}$$

$$c'(q,D) = \frac{c(q,D)}{1-b+b\frac{|D|}{avdl}}$$

The End