复旦大學
FUDAN UNIVERSITY

文本表示与相似度

邱锡鹏
复旦大学
http://nlp.fudan.edu.cn/xpqiu

需求

- 判断两段文本的相似度或相关度
  - 搜索引擎：查询与文档（网页）
  - 新闻聚类：文档与文档
  - 重复网页检测
  - 抄袭检测

文本表示与相似度计算

- 相似度和距离
  - 距离d(A,B)
    - [0,∞]
  - 相似度s(A,B)
    - [0,1]
  - 距离和相似度可以相互转换

文本表示与相似度计算

- 给定两段文本，如何判断它们之间的相似度
- 表示形式
  - 符号表示
  - 向量表示
- 相似度
  - 浅层相似度 surface similarity
  - 结构相似度 structure similarity
  - 语义相似度 semantic similarity

符号化表示与词汇相似度计算

基于符号表示的相似度计算

- 用途
  - 聚类
  - 去冗余
  - 信息检索

基于符号表示的相似度计算

- 粒度 Granularity
  - 字符
    - 字符串匹配
    - 编辑距离
  - 词
    - Jaccard相似度
  - 短语
  - 句子
  - 文档

基于符号表示的相似度计算

- 编辑距离
- Jaccard相似度
- 最小哈希 MinHash

编辑距离 Edit Distance

- 编辑距离：给定2个字符串 a, b. 编辑距离是将 a 转换为 b 的最少操作次数。
- Levenshtein距离是编辑距离的一种，允许的操作包括一个字符替换成另一个字符，插入一个字符，删除一个字符。
  - 例如，将kitten一字转成sitting:
    - sitten （k→s）
    - sittin （e→i）
    - sitting （→g）

编辑距离-算法

- 用分治的思想解决比较简单，将复杂的问题分解成相似的子问题
  - 假设字符串 a, 共 m 位，从 a[1] 到 a[m]
  - 字符串 b, 共 n 位，从 b[1] 到 b[n]
  - d[i][j] 表示字符串 a[1]-a[i] 转换为 b[1]-b[i] 的编辑距离

编辑距离-算法

- 递归边界
  - a[i][0] = i, b 字符串为空，表示将 a[1]-a[i] 全部删除，所以编辑距离为 i
  - a[0][j] = j, a 字符串为空，表示 a 插入 b[1]-b[j]，所以编辑距离为 j

编辑距离-算法

- 那么有如下递归规律：
  - 当 a[i] 等于 b[j] 时，d[i][j] = d[i-1][j-1]
    - 比如 fxy -> fay 的编辑距离等于 fx -> fa 的编辑距离
  - 当 a[i] 不等于 b[j] 时，d[i][j] 等于如下 3 项的最小值：
    - d[i-1][j] + 1（删除 a[i]）
      - 比如 fxy -> fab 的编辑距离 = fx -> fab 的编辑距离 + 1
    - d[i][j-1] + 1（插入 b[j]）
      - 比如 fxy -> fab 的编辑距离 = fxyb -> fab 的编辑距离 + 1 = fxy -> fa 的编辑距离 + 1
    - d[i-1][j-1] + 1（将 a[i] 替换为 b[j]）
      - 比如 fxy -> fab 的编辑距离 = fxb -> fab 的编辑距离 + 1 = fx -> fa 的编辑距离 + 1

编辑距离-算法

- 初始化

  D(i,0) = i

  D(0,j) = j

- 递归

  For each i = 1…m

  For each j = 1…n

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} r; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

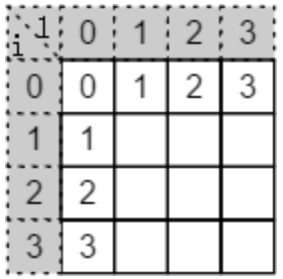
  r = 1 or 2

- 终止

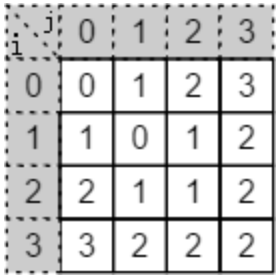
  D(m,n)

示例

- 以字符串 a = "fxy", b = "fab “
  - 首先建立一个矩阵，用来存放子问题及原问题的编辑距离，并将递归边界在矩阵中填好。
  - 然后计算 i = 1, j = 1 所对应的编辑距离：比较 a[i] 和 b[j] 是否相等然后根据递归规律算出这个值。
    - 比如在这种情况下 a[i] = f 和 b[j] = f, 那么 d[i][j] 就等于 d[i-1][j-1] 等于 0
    - 然后计算 i = 1, j = 2 直到算出 i = 3, j = 3, 原问题的编辑距离就等于 d[3][3]



i \ j	0	1	2	3
0	0	1	2	3
1	1			
2	2			
3	3			



i \ j	0	1	2	3
0	0	1	2	3
1	1	0	1	2
2	2	1	1	2
3	3	2	2	2

加权编辑距离

- 加权
  - 拼写检查：一些字母更容易打错

Weighted Min Edit Distance

- 初始化

$$D(0,0) = 0$$
$$D(i,0) = D(i-1,0) + del[x(i)]; \quad 1 < i \leq N$$
$$D(0,j) = D(0,j-1) + ins[y(j)]; \quad 1 < j \leq M$$

- 递归

$$D(i,j) = \min \begin{cases} D(i-1,j) + del[x(i)] \\ D(i,j-1) + ins[y(j)] \\ D(i-1,j-1) + sub[x(i),y(j)] \end{cases}$$

- 终止

$$D(m,n)$$

拼写校正

- Damerau-Levenshtein距离
  - 插入
  - 删除
  - 替换
  - 邻近字符转换

Words within 1 of acress

Error	Candidate Correction	Correct Letter	Error Letter	Type
acress	actress	t	-	deletion
acress	cress	-	a	insertion
acress	caress	ca	ac	transposition
acress	access	c	r	substitution
acress	across	o	e	substitution
acress	acres	-	s	insertion
acress	acres	-	s	insertion

候选词生成

- 80%错误的编辑距离为1
- 几乎所有错误的编辑距离在2以内

- 允许加入空格和连接符（hyphen）
  - thisidea → this idea
  - inlaw → in-law

编辑距离

- 应用
  - DNA分析
  - 拼写检查
  - 语音辨识
  - 抄袭侦测

Jaccard相似度

- 集合：无序的对象的组合
  - 元素之间是无序的，地位相同的。
  - 任何两个元素都认为是不同的，每个元素只能出现一次。
  - 给定一个集合，任给一个元素，该元素或者属于或者不属于该集合，二者必居其一，不允许有模棱两可的情况出现。

Jaccard相似度

- 计算两个集合的相似度
  - $J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$
  - $0 \leq J(A,B) \leq 1$

文档到集合

- 词袋模型（Bag-of-Words）
  - 一篇文档可以看作是一个无序的词的集合。

- N元模型
  - 词袋模型的泛化
  - 连续的n个词作为集合中的一个元素

I like apple.

基于Jaccard系数的文本相似度

- I like apple.
- I do not like apple.

技巧

- 空格、间隔符
- 大小写
- 标点符号
- 粒度：字符、词
- N取值
- 停用词

最小哈希 MinHash

- 最小哈希方法是一种快速判断两个集合是否相似的技术。
  - 是一种局部性敏感哈希
  - 由Andrei Broder (1997) 发明并用于在搜索结果中检测并消除重复Web页面。
  - 也应用于大规模聚类问题，比如通过文档间包含的词语相似性进行聚类。

1. Broder, Andrei Z. "On the resemblance and containment of documents." Compression and Complexity of Sequences 1997. Proceedings. IEEE, 1997.
2. Broder, Andrei Z.; Charikar, Moses; Frieze, Alan M.; Mitzenmacher, Michael, Min-wise independent permutations, Proc. 30th ACM Symposium on Theory of Computing (STOC '98), 1998.

MinHash

- MinHash目标是快速估计 J(A,B) 的值，不用计算without explicitly computing the 交和并得操作。
- h是一个哈希函数,将A、B的元素映射到不同的整数。
- $h_{min}(S)$是S中的元素x，其h(x)的值最小。
- 只有当并集A∪B中具有最小哈希值的元素属于交集A∩B时，有$h_{min}(A) = h_{min}(B)$。
- $Pr[\ h_{min}(A) = h_{min}(B)\ ] = J(A,B)$,

MinHash

Element	S_1	S_2	S_3	S_4
2	1	0	1	0
5	1	0	0	0
6	0	0	1	1
1	1	0	0	1
4	0	0	1	1
3	0	1	1	0

- 随机排序（按行）
- 记录每一列的第一个1
  - $m(S_1)=2$
  - $m(S_2)=3$
  - $m(S_3)=2$
  - $m(S_4)=6$
- 估计Jaccard相似度
  - $JS(Si, Sj)=\begin{cases}1 & m(Si)=m(Sj)\\ 0 & otherwise\end{cases}$

语义相似度

- 语义相关
  - 会议几点开始?
  - 什么时间开会?
- 语义不相关
  - 要他的狗命
  - 要他狗的命

多词一义和一词多义

- 词的同义现象和多义现象是自然语言处理中最基本的问题。
  - 同义现象指不同的词有相同的意思。
    - 多词一义
    - doctor physicians
    - car auto
  - 多义现象指一个词具有多种不同的意思。
    - 一词多义
    - doctor/apple/tree/bank/fox

引入语义词典

- WordNet
- 同义词词林
- 知网
- …

WordNet

https://wordnet.princeton.edu/

- WordNet是一部英语语义词典
  - 免费的在线词汇数据库
  - 根据词义而不是词形来组织词汇信息
  - 由普林斯顿大学心理语言学实验室开发
  - 在自然语言处理中得到广泛的应用

Demo: http://wordnetweb.princeton.edu/perl/webwn

search “tree” or “bank”

WordNet

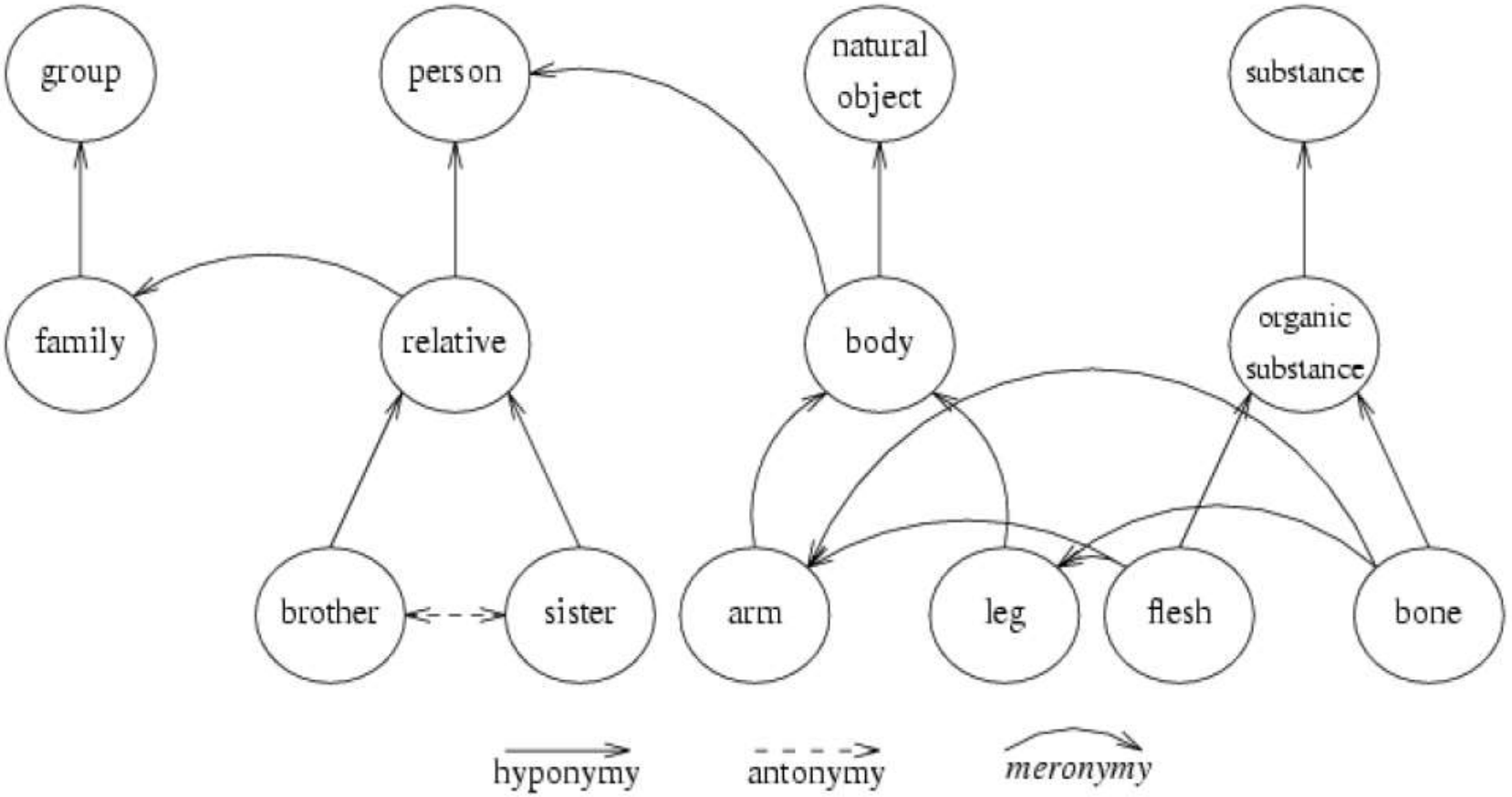
- 同义词集Synset
  - 用一组同义词的集合Synset来表示一个概念
  - 每一个概念有一段描述性的说明
- 同义词集之间的关系
  - 上下位关系（hypernymy，hyponymy）
    - rabbit/animal
  - 反义关系（antonymy）
    - good/bad
  - 部分整体关系（部分词（meronym）/整体词（holonym））
    - leaf/tree

WordNet 3.0统计

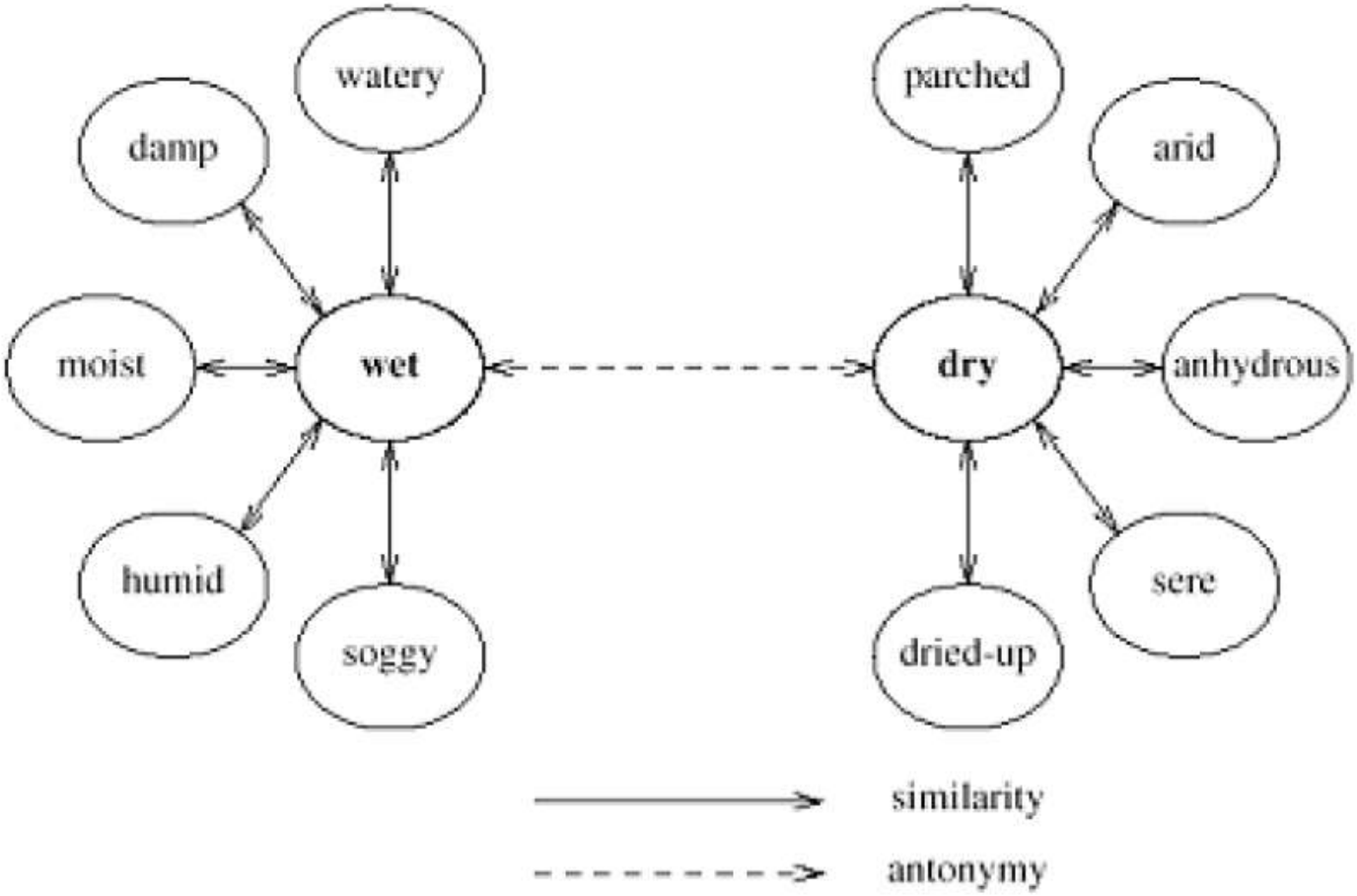
Number of words, synsets, and senses

POS	Unique Strings	Synsets	Total Word-Sense Pairs
Noun	117798	82115	146312
Verb	11529	13767	25047
Adjective	21479	18156	30002
Adverb	4481	3621	5580
Totals	155287	117659	206941

WordNet示例


group
person
natural object
substance
family
relative
body
organic substance
brother
sister
arm
leg
flesh
bone
hyponymy
antonymy
meronymy

WordNet示例


watery
damp
parched
arid
moist
wet
dry
anhydrous
humid
soggy
dried-up
sere
similarity
antonymy

同义词词林

- 中文同义词词典
  - 不仅包括了一个词语的同义词，也包含了一定数量的同类词，即广义的相关词
  - 梅家驹等，1983，上海辞书出版社
  - 初衷是希望提供较多的同义词，以克服写作和翻译时的词穷现象而编写
  - 目前广泛应用于自然语言处理中
  - 收词近7万（按义项统计）
- 按义项编排
  - 12大类
  - 94中类
  - 1428小类
  - 3925词群
  - 词群内部的词是同义词
  - 大类、中类、小类之间不一定是上下位关系（有些是领域）

同义词词林

Ag100101	旅客
Ag100101	客人
Ag100101	旅人
Ag100101	客子
Ag100101	客行子
Ag100101	游子
Ag100101	行人
Ag100101	行者
Ag100101	行旅
Ag100101	行客
Ag100101	行子
Ag100101	征人
Ag100101	征夫
Ag100101	征客
Ag100101	羁客
Ag100101	羁旅
Ag100101	客
Ag100102	过路人
Ag100102	过客
Ag100103	游人
Ag100103	游客
Ag100103	游者
Ag100103	旅游者
Ag100103	观光者

大类：A

中类：g

小类：10

词群：01

最小同义词集：01，02，03

知网（Hownet）

http://www.keenage.com

- 知网（英文名称为HowNet）是一个以汉语和英语的词语所代表的概念为描述对象，以揭示概念与概念之间以及概念所具有的属性之间的关系为基本内容的常识知识库。

- 概念描述举例

  NO.=017144
  W_C=打
  G_C=V
  E_C=~网球，~牌，~秋千，~太极，球~得很棒
  W_E=play
  G_E=V
  E_E=
  DEF=exercise|锻练,sport|体育

- 其中DEF是核心，采用特定的“知识描述语言”

知网（Hownet）

- 义原的上下位关系构成树结构

```
- entity | 实体
 ├ thing | 万物
 … ├ physical | 物质
   … ├ animate | 生物
     … ├ AnimalHuman | 动物
       … ├ human | 人
         |  └ humanized | 拟人
         └ animal | 兽
           ├ beast | 走兽
       …
```

基于《知网》的词汇语义相似度计算

- 词语相似度计算
  - 对于两个词语W1和W2，如果W1有n个义项（概念）：S11，S12，……，S1n，W2有m个义项（概念）：S21，S22，……，S2m，
  - W1和W2的相似度各个概念的相似度之最大值

$$Sim(W_1, W_2) = \max_{i=1..n, j=1...m} Sim(S_{1i}, S_{2j})$$

刘群, & 李素建. (2002). 基于《知网》的词汇语义相似度计算. 中文计算语言学, 7(2), 59-76.

基于《知网》的词汇语义相似度计算

- 义原相似度计算
  - 所有的义原根据上下位关系构成了一个树状的义原层次体系
  - 假设两个义原在这个层次体系中的路径距离为d，可以得到这两个义原之间的语义距离：

$$Sim(p_1, p_2) = \frac{\alpha}{d + \alpha}$$

缺点?
需要考虑义原的深度!

刘群, & 李素建. (2002). 基于《知网》 的词汇语义相似度计算. 中文计算语言学, 7(2), 59-76.

基于知识库的语义相似度计算

- 知识库一般可以看作为图或树结构
- 对于两个词，它们的相似度可以根据下面策略计算：
  - 是否为同义词
  - 词在知识库中的最短路径
    - 不同类型的边可以赋予不同的权重

思考

- 怎样计算短语、句子、篇章的语义相似度?

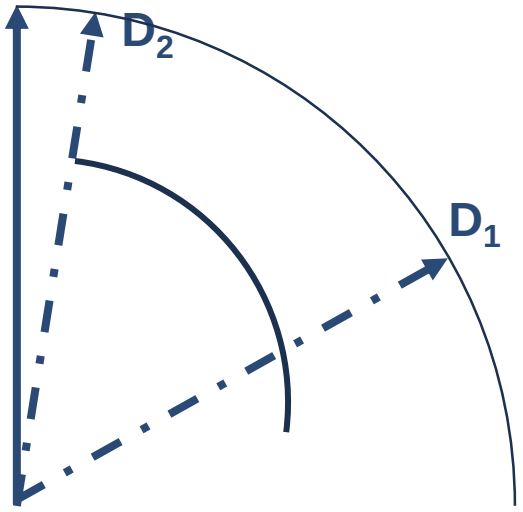
基于向量表示的语义相似度

基于向量的文本表示

- 向量表示
  - 用向量来表示文本
    - 词
    - 句子
    - 文档
- 假设文本t1,t2对应的向量表示为v1,v2,则t1,t2的相似度可以为
  - 欧式距离 $\mathrm{sim}(t_1, t_2) = \|v_1 - v_2\|^2$
  - 内积 $\mathrm{sim}(t_1, t_2) = v_1^T v_2$
  - Cosine距离 $\mathrm{sim}(t_1, t_2) = \mathrm{cosine}(v_1, v_2)$

Cosine相似度

- 两个向量的夹角
  - $cosine(V_1, V_2) = \frac{V_1^{\mathrm{T}} V_2}{|V_1|_2 \times |V_2|_2}$



向量空间模型

- 向量空间模型，Vector Space Model，VSM
  - 信息检索
- 用向量来表示文档和查询
  - 每一维表示一个概念
    - 词、主题
  - 每一维的值代表概念的权重
    - E.g., $d=(x_1,\cdots,x_k)$, x_i 表示概念i的重要性
- 相似度
  - 文档和查询的向量表示的Cosine距离

什么是好的“概念”？

- 概念之间相互独立
  - 不重叠、没有歧义
- 权重方便计算

- 常见的“概念”
  - 词项 Terms
  - N-Grams
  - 主题

词频 Term Frequency，TF

- 词频（term frequency，TF）指的是某一个给定的词语在该文档中出现的频率。
- 长文档
  - 罗嗦
  - 更多内容
- 原始TF是不准确的
  - 归一化
  - $tf_{ij} = \frac{n_{ij}}{\sum_k n_{k,j}}$
  - n_{ij}是该词在d_j中的出现次数

逆向文档频率 Inverse Document Frequency，IDF

- 逆向文档频率（inverse document frequency，IDF）是一个词语普遍重要性的度量。
- 某一特定词语的IDF，可以由总文档数目除以包含该词语之文件的数目，再将得到的商取对数得到.
  - $IDF(t_i) = 1 + \log(\frac{N}{df(t_i)})$
  - 包含词语 t_i 的文件数目
- 整个数据集相关，和单个文档无关

TF-IDF

- 信息检索的常见模型 (G Salton et al. 1983)
- 结合TF和IDF
  - Common in doc → high tf → high weight
  - Rare in collection→ high idf→ high weight
  - $w(t,d) = TF(t,d) \times IDF(t)$

"Salton was perhaps the leading computer scientist working in the field of information retrieval during his time." - wikipedia

Gerard Salton Award – highest achievement award in IR

TF-IDF不足

- 词项独立性假设
- 权重计算
  - 突出重要单词，抑制次要单词
  - 不完全正确
- 位置信息

词-文档矩阵

- 词-文档矩阵
  - 或共现矩阵（Occurences Matrix）
- 词-文档矩阵式一个稀疏矩阵，其行代表词语，其列代表文档。元素为
  - 该词在文档中的是否出现
  - 该词在文档中的出现次数
  - 该词语的TF-IDF

- 词-文档矩阵是词袋模型、向量空间模型的“矩阵”表示形式。

例子

http://web.stanford.edu/class/linguist289/lsi.pdf

Technical Memo Example

human computer interaction

Titles:

c1: *Human* machine *interface* for Lab ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management *system*
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user*-perceived *response time* to error measurement

m1: The generation of random, binary, unordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

graphs

例子

http://web.stanford.edu/class/linguist289/lsi.pdf

Technical Memo Example

human computer interaction

Titles:

c1: *Human* machine *interface* for Lab ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management *system*
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user*-perceived *response time* to error measurement

m1: The generation of random, binary, unordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

graphs

例子

http://web.stanford.edu/class/linguist289/lsi.pdf

Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

词-文档矩阵的两种视角

- 词-文档矩阵 X
  - X[i,:]表示第i个词
  - X[:,j]表示第j个文档

- XX^T ?
- X^TX ?

显示语义分析
Explicit Semantic Analysis，ESA

- 用大规模的文本数据集来增强词袋模型
  - Wikipedia
- 构建词-文档矩阵 X
  - 用X[i,:]表示第i个词
- 对于文本T，向量表示（IFIDF）为v
  - v为m维向量
- 其ESA表示为

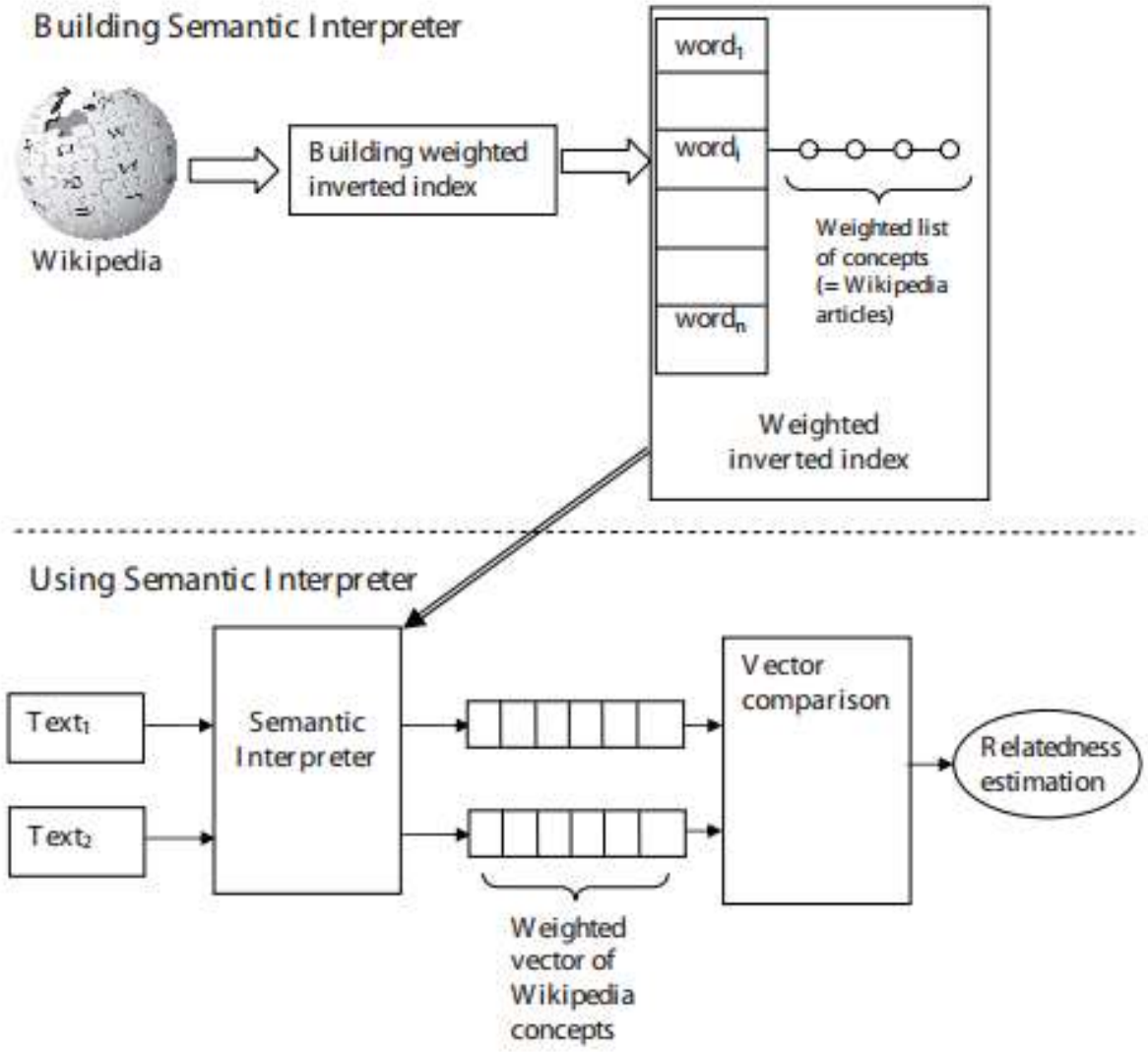
$$v' = \sum_{i=1}^{m} v_m \, X[i,:]$$

ESA实现技巧

- 数据稀疏性问题
  - 去掉高频词
  - 不用向量表示，用倒排表

流程

Gabrilovich, Evgeniy; Markovitch, Shaul (2007). Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis . IJCAI



示例

Gabrilovich, Evgeniy; Markovitch, Shaul (2007). Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis . IJCAI

#	Ambiguous word: "Bank"		Ambiguous word: "Jaguar"	
	"Bank of America"	*"Bank of Amazon"*	*"Jaguar car models"*	*"Jaguar (Panthera onca)"*
1	Bank	Amazon River	Jaguar (car)	Jaguar
2	Bank of America	Amazon Basin	Jaguar S-Type	Felidae
3	Bank of America Plaza (Atlanta)	Amazon Rainforest	Jaguar X-type	Black panther
4	Bank of America Plaza (Dallas)	Amazon.com	Jaguar E-Type	Leopard
5	MBNA	Rainforest	Jaguar XJ	Puma
6	VISA (credit card)	Atlantic Ocean	Daimler	Tiger
7	Bank of America Tower, New York City	Brazil	British Leyland Motor Corporation	Panthera hybrid
8	NASDAQ	Loreto Region	Luxury vehicles	Cave lion
9	MasterCard	River	V8 engine	American lion
10	Bank of America Corporate Center	Economy of Brazil	Jaguar Racing	Kinkajou

词-文档矩阵的不足

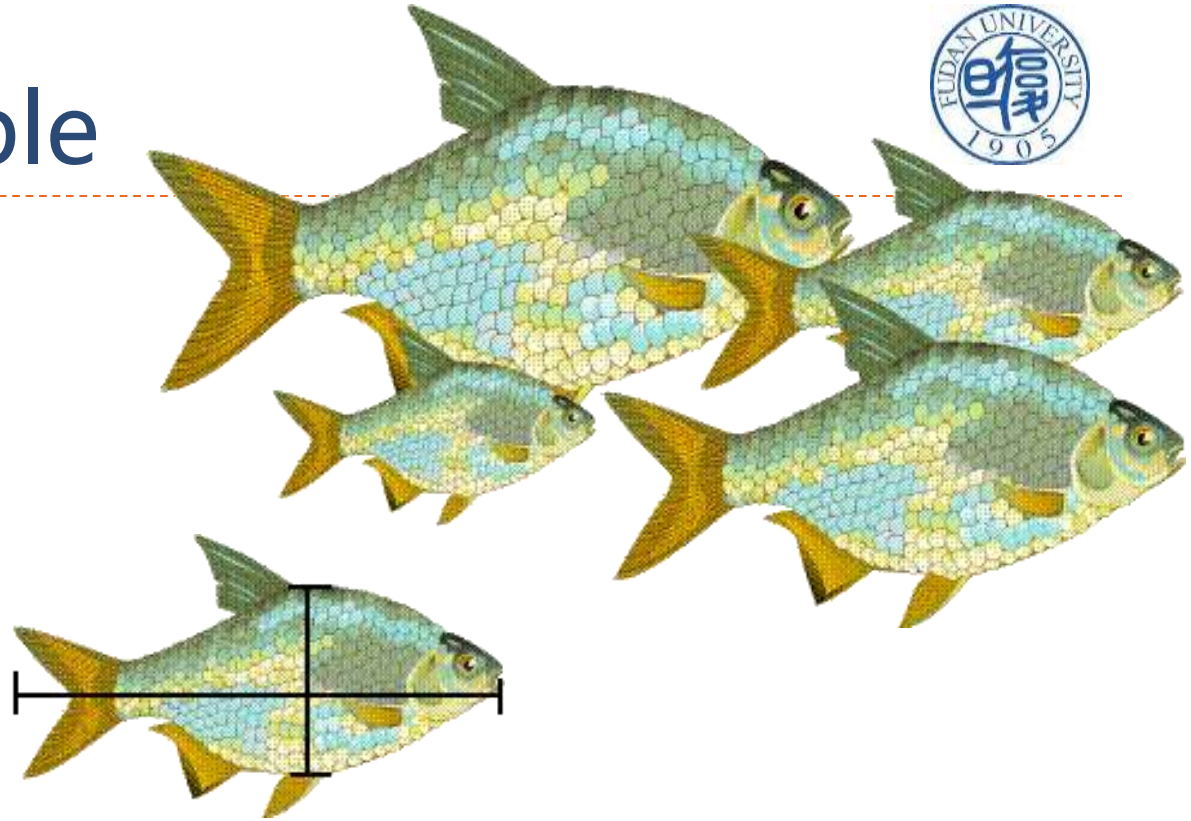
- 词-文档矩阵 X
  - X[i,:]表示第i个词
  - X[:,j]表示第j个文档

- 词之间是相互独立的
  - XX^T 是对角阵
- 文档之间是相互独立的
  - X^TX 是对角阵

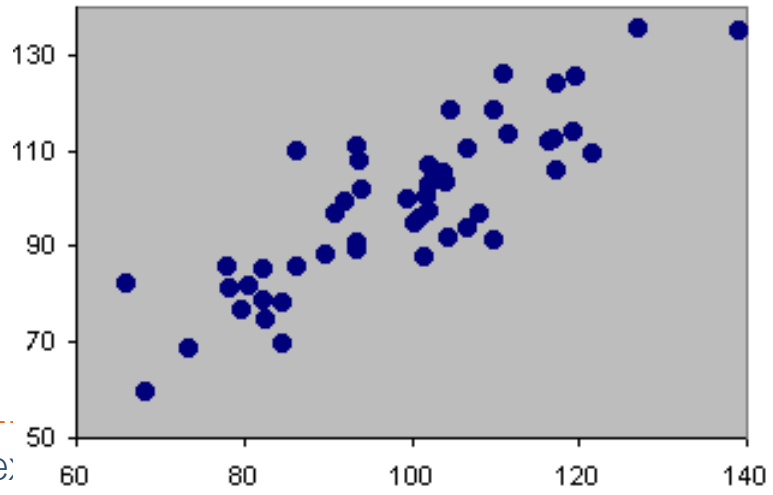
不成立！
如何找到相互独立的基向量？

Visual Example

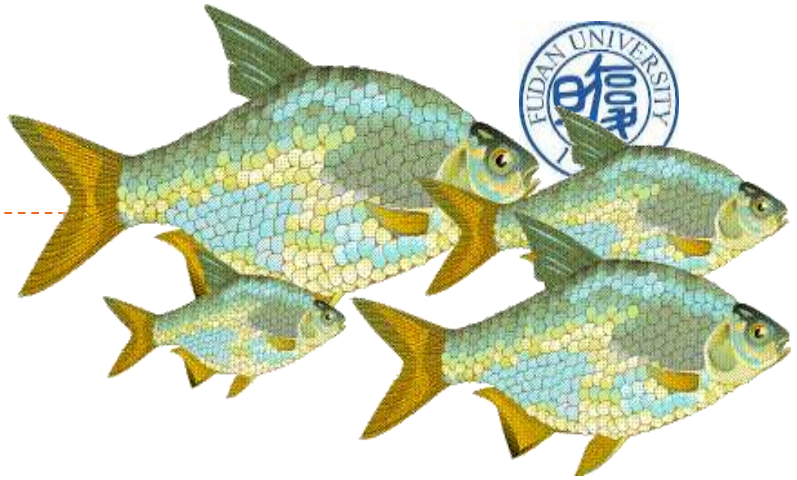
- Data on Fish
  - Length
  - Height



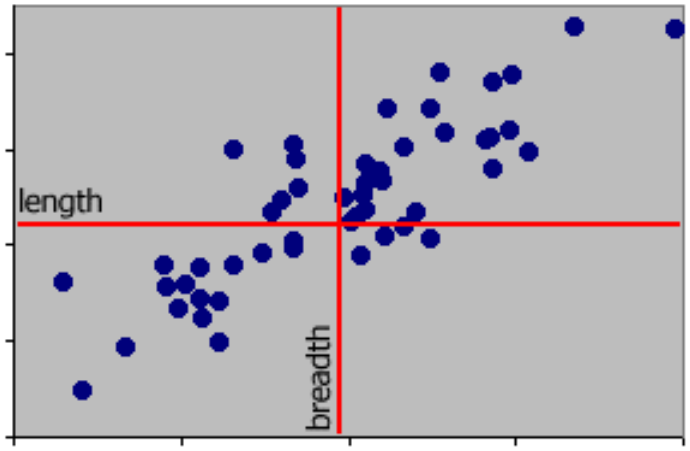
Slides from
http://rakaposhi.eas.asu.edu/cse494/

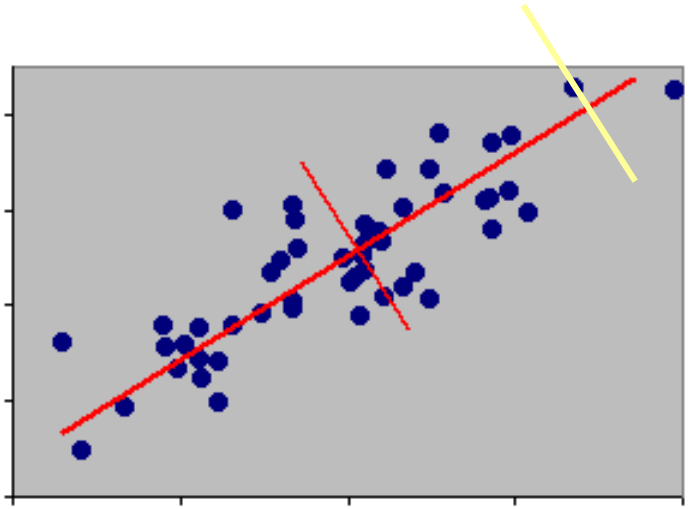


Move Origin



- To center of centroid
- But are these the best axes?





Better if one axis accounts for most data variation and each axis is orthogonal to the others

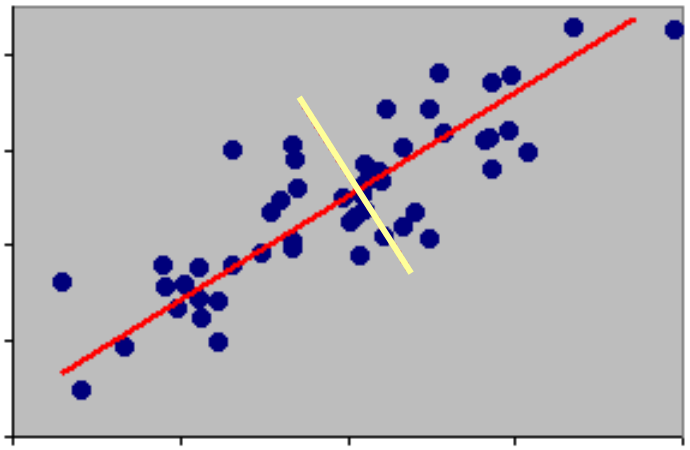
What should we call the red axis? “Size” (“factor”)

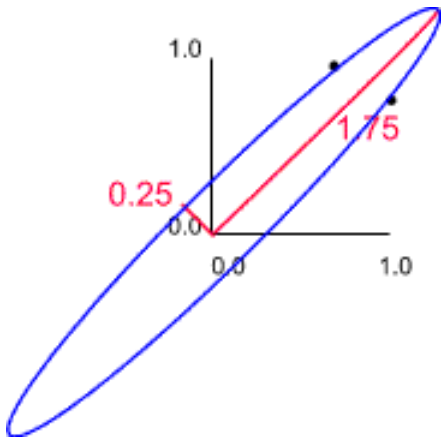
Reduce Dimensions

- What if we *only* consider "size"

We retain 1.75/2.00 x 100 (87.5%)
of the original variation.

Thus, by discarding the yellow axis
we lose only 12.5%
of the original information.





数学基础

- 一组随机向量 $X=[x_1, x_2, \cdots, xn], x_i \in Rm$
- 数学期望

$$E(x) = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- 协方差

$$\mathrm{cov}(x^{ij}) = E[(x^i - E(x^i))(xj - E(x^j))]$$

- 假设$E(x) = 0$ $\mathrm{cov}(x^{ij}) = \frac{1}{n} X[i,:]^T X[j,:]$

- 协方差矩阵 $C = \frac{1}{n} XX^T$

主成分分析 PCA

- 数据降维的重要手段之一
  - 去除冗余、噪声
- 坐标变换 $X' = P^T X$
  - P是标准化矩阵，由于$P^{-1}P=I$，$P^TP=I$
- 使得$X'X'^T = \Lambda$，即 $P^T(XX^T)P = \Lambda$

$$XX^T = P\Lambda P^T$$

  - Λ为对角阵
  - P为协方差矩阵的特征向量

矩阵分解

- Singular value decomposition（SVD）
- 对于任意m×n的输入矩阵X，SVD分解结果为：
  - $X_{[m\times n]}=U_{[m\times m]}\,\Sigma_{[m\times n]}\,(V_{[n\times n]})^{T}$
    - U为左奇异向量（left singular vectors）
    - Σ为奇异值矩阵
    - V为右奇异向量（right singular vectors）
    - 矩阵U, V中的列向量均为正交单位向量，而矩阵Σ为对角阵，并且从左上到右下以递减的顺序排序。

例子

- https://zh.wikipedia.org/wiki/奇异值分解

观察一个4×5的矩阵

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \end{bmatrix}.$$

M矩阵的奇异值分解如下$U\Sigma V^*$

$$U = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \Sigma = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, V^* = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ \sqrt{0.8} & 0 & 0 & 0 & -\sqrt{0.2} \end{bmatrix}.$$

SVD与PCA的关系

- SVD

$$XX^T = U\Sigma V^T (U\Sigma V^T)^T = U\Sigma V^T V\Sigma U^T = U\Sigma^2 U^T$$

- PCA

$$XX^T = P\Lambda P^T$$

对文档进行降维

- 文档向量
  - 每一维对于一个词
  - 具有冗余性
- 找到新的子空间，使得
  - 每一维是独立的，正交的
  - 选取前K维，以减少噪声
- 方法
  - 对词-文档矩阵进行SVD分解

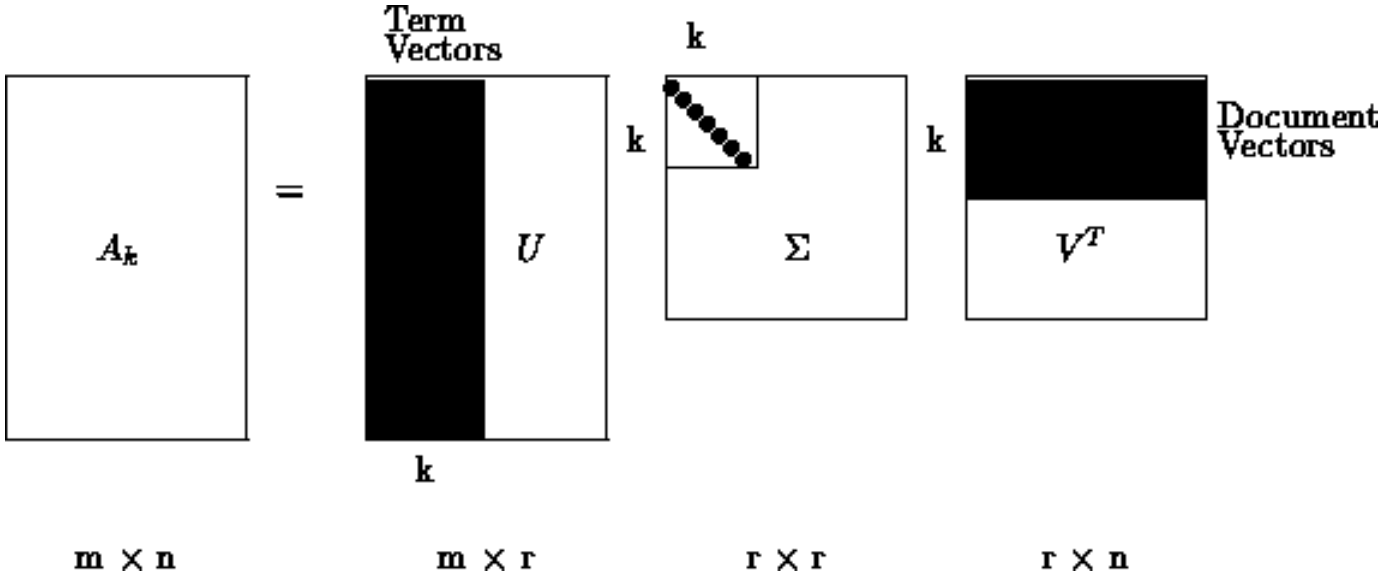
潜在语义分析

潜在语义分析

- 潜在语义分析
  - Latent Semantic Analysis，LSA
  - 潜在语义索引 Latent Semantic Indexing，LSI

- 转换后的每一维代表一个“潜在概念”
  - Latent Concept

潜在语义分析



Term x Document Matrix

Term x Concept Matrix

Concept Matrix

Concept x Document Matrix

潜在语义分析

- SVD

$$X \approx \bar{X} = U\Sigma V^T$$

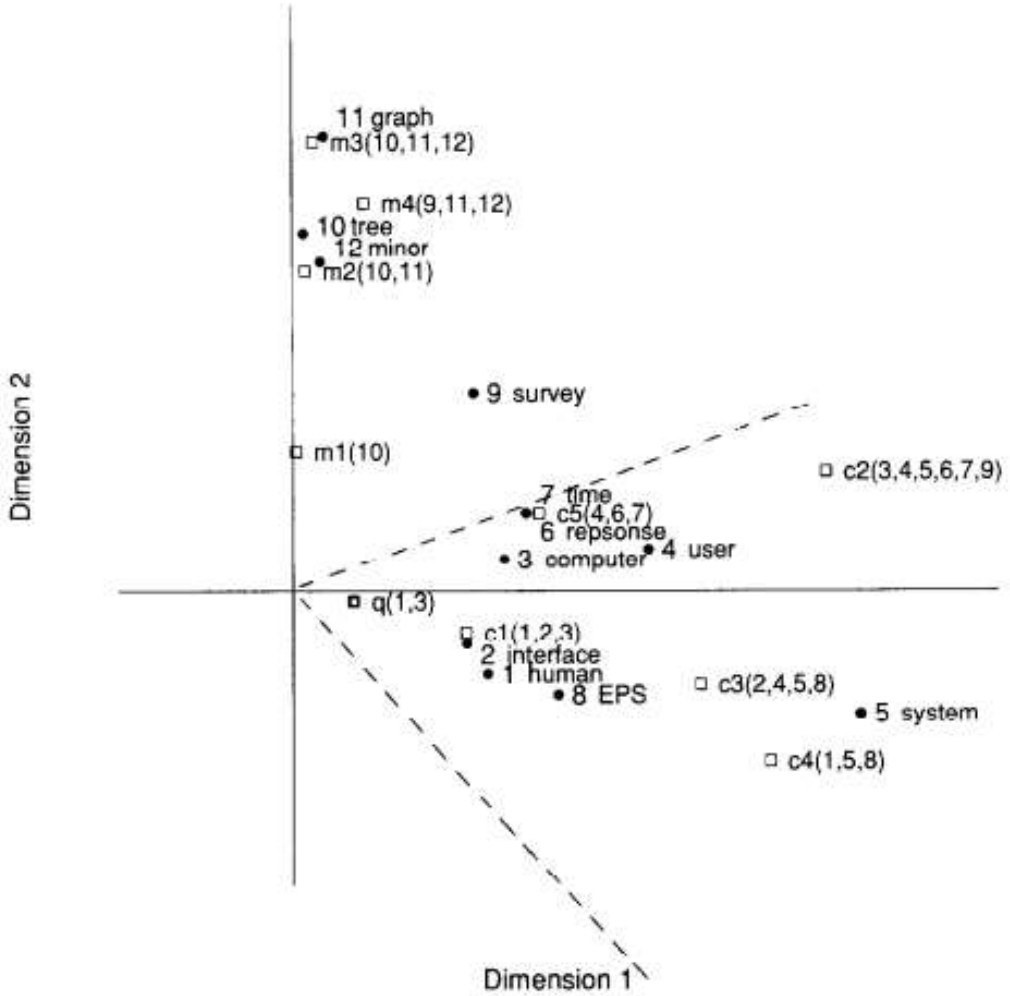
- 新的文档向量

$$\bar{d} = U^T d$$

- 新的文档向量

$$\bar{t} = V^T t$$

2-D Plot of Terms and Docs from Example



LSA分析

- 低维的语义空间可以用于以下几个方面:
  - 在低维语义空间可对文档进行比较，进而可用于文档聚类和文档分类。
  - 在翻译好的文档上进行训练，可以发现不同语言的相似文档，可用于跨语言检索。
  - 发现词与词之间的关系，可用于同义词、歧义词检测。.
  - 通过查询映射到语义空间，可进行信息检索。

LSA分析

- SVD分解复杂度高 $O(n^3)$
- LSA只能去除线性相关冗余性
  - $x^2+y^2=1$

- 改进
  - 概率潜在语义分析
  - 潜在狄利克雷分析　　后面会细讲
  - 分布式语义表示

其它方法

Pointwise Mutual Information (PMI)

- 互信息

$$\text{PMI}(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)}$$

- Positive PMI

$$\text{PPMI}(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P(c)}, 0)$$

Normalized Google Distance (NGD)

- 给定两个词x，y

$$\mathrm{NGD}(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log N - \min\{\log f(x), \log f(y)\}}$$

- N 所有的网页数
- f(x) 和f(y)分别是搜索x和y的网页数
- f(x, y)是x和y一起出现的网页数

https://arxiv.org/abs/cs/0412098

总结

https://en.wikipedia.org/wiki/Semantic_similarity

- 主要是基于词的相似度
  - 符号表示可以转换为向量表示
  - 忽略了句子的结构、语法信息

	字面相似度	语义相似度
符号表示	编辑距离 Jaccard相似度 最小哈希 MinHash	知网 HowNet
向量表示	向量空间模型VSM	显示语义分析ESA 潜在语义分析LSA

谢 谢

如果您有任何意见、评论以及建议，请通过GitHub的Issues页面进行反馈。