数据挖掘导论
Introduction to Data Mining

第五章 文本挖掘

刘 淇
Email: qiliuql@ustc.edu.cn
课程主页：
http://staff.ustc.edu.cn/~qiliuql/DM2017YZ.html

目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的常用方法
    - 主题模型
    - word2vec

文本挖掘

- 在当今数据规模飞速增长的时代，文本数据分析面临着巨大的挑战。
- 对象复杂
  - 网页信息、论文期刊、新闻文章、电子书籍等等。
  - 具有标题、作者、出版时间等结构化的信息，也有由大量自然语言描述的文本组成。
- 数据量庞大
  - 西方四大通讯社每天总发稿量高达3500万字。
  - 全世界每年发表的科学技术文献达400万件以上，而且每年还以5%～7%的速度增长。
  - 网站互相链接，仅与美国网络公司雅虎的链接站点就有75万个。

文本挖掘

- 传统的自然语言理解技术主要进行基于词、语法和语义信息的分析，并通过词在句子中出现的次序发现有意义的信息。
  - 这种方法面对海量的文本数据以及对整个段落或文章的理解，无从下手。
- 文本挖掘的方法高层次理解的对象可以是仅包含简单句子的单个文本，也可以是多个文本组成的文本集，从而分析文本的模式、相关性、差异性等。

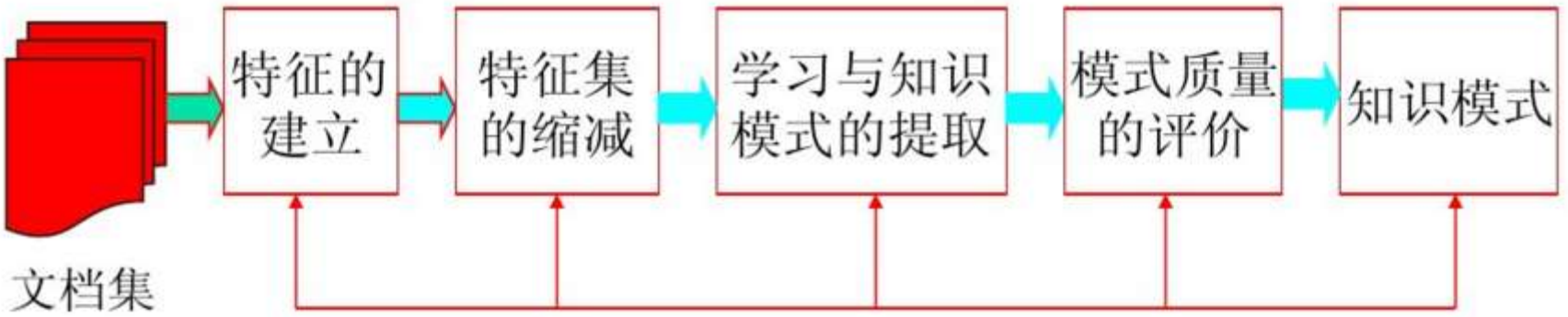
文本挖掘概念

- 文本挖掘
  - 旨在通过识别和检索令人感兴趣的模式，进而从数据源中抽取有用的信息。
  - 文本挖掘的数据源通常是文本集合，令人感兴趣的模式不是从形式化的数据库记录里发现，而是从非结构化的数据中发现

文本挖掘的任务

- 文本挖掘预处理
  - 原始的非结构化数据源 → 结构化表示
- 文本模式挖掘
  - 文本挖掘系统核心功能是分析文本集合中各个文本之间共同出现的模式
  - 例如：蛋白质P1和酶E1存在联系，在其他文字中说酶E1和酶E2的功能相似，还有文章将酶E2和蛋白质P2联系起来，那么可以推断出P1和P2存在联系
- 挖掘结果可视化
  - 也就是文本挖掘系统的表示层，简称浏览

文本挖掘的处理过程

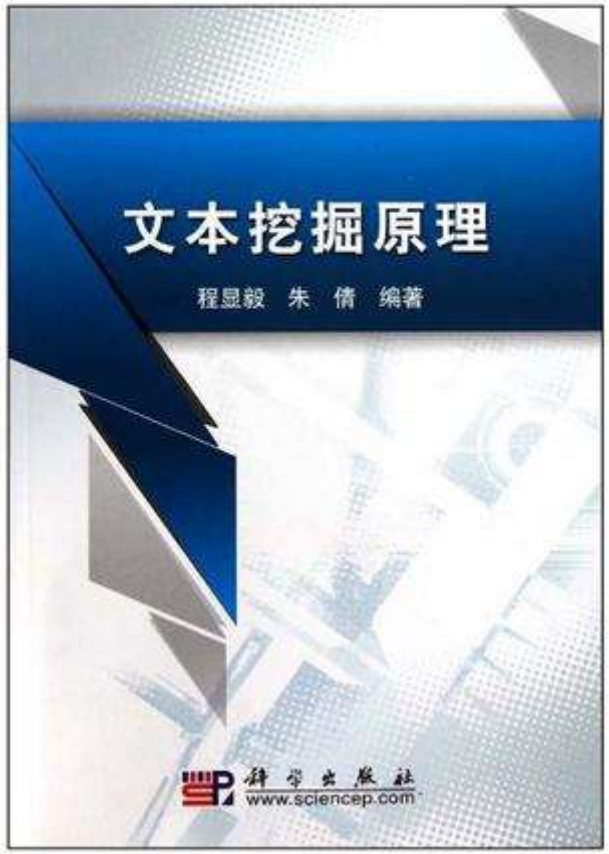


文本挖掘的一般处理过程

文本挖掘

- 《文本挖掘原理》
  - 作者: 程显毅 / 朱倩
  - 出版社: 科学出版社

- 《The Text Mining Handbook》
  - 作者: Ronen Feldman / James Sanger
  - 出版社: Cambridge University Press



目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的常用方法
    - 主题模型
    - word2vec

文本特征抽取

- 定义：文本特征指的是关于文本的元数据
- 分类：
  - 描述性特征：文本的名称、日期、大小、类型等
  - 语义性特征：文本的作者、标题、机构、内容等

很辛苦，但同时也是特别重要的一个环节

特征抽取（Feature Extraction）

- 预处理
  - 去掉html一些tag标记
  - 禁用词(stop words)去除、词根还原(stemming)
  - (中文)分词、词性标注、短语识别、…
  - 词频统计
    - **$TF_{i,j}$: 特征i在文档j中出现次数，词频(Term Frequency)**
    - **DF_i:所有文档集合中出现特征i的文档数目，文档频率(Document Frequency)**
  - 数据清洗：去掉不合适的噪声文档或文档内垃圾数据
- 文本表示
  - 向量空间模型
- 降维技术
  - 特征选择(Feature Selection)
  - 特征重构(Re-parameterisation，如LSI)

文本表示

- 向量空间模型(Vector Space Model)
  - M个无序标引项t_i (词条项, 特征)，词根/词/短语/其他
  - 每个文档d可以用标引项向量来表示

$$V(d) = (t_1, w_1(d); \ldots; t_i, w_i(d); \ldots; t_n, w_n(d))$$

  - 权重计算，N个训练文档
    - $W_{M*N} = (w_{ij})$
  - 词项的权重：{0, 1}， tf(词频=term frequency)，tf*idf （ **$w_{ij} = TF_{ij} * \log(N/DF_i)$** ）

文本表示

- 词频矩阵（在VSM的基础上）
  - 行对应关键词t，列对应文档d向量
  - 将每一个文档视为空间向量v
  - 向量值反应单词t与文档d的关联度
  - 矩阵元素可以是词频，也可以是布尔型

表示文档词频的词频矩阵

	d_1	d_2	d_3	d_4	d_5	d_6
t_1	322	85	35	69	15	320
t_2	361	90	76	57	13	370
t_3	25	33	160	48	221	26
t_4	30	140	70	201	16	35

中文特征词（Term）的粒度

- Character，字：中
- Word，词：中国
- Phrase，短语：中国人民银行
- Concept，概念：
- 同义词：开心、高兴、愉快
- 相关词cluster，word cluster：葛非/顾俊
- N-gram，N元组：中国 国人 人民 民银 银行
- 某种规律性模式：比如某个window中出现的固定模式

主要的分词方法

- 基于词典的分词方法
  - 最大匹配法（Maximum Matching method, MM法）：选取包含6-8个汉字的符号串作为最大符号串，把最大符号串与词典中的单词条目相匹配，如果不能匹配，就削掉一个汉字继续匹配，直到在词典中找到相应的单词为止。匹配的方向是从右向左。
  - 逆向最大匹配法（Reverse Maximum method, RMM法）：匹配方向与MM法相反，是从左向右。实验表明：**对于汉语来说，逆向最大匹配法比最大匹配法更有效。**
  - 双向匹配法（Bi-direction Matching method, BM法）：比较MM法与RMM法的分词结果，从而决定正确的分词。
  - 最佳匹配法（Optimum Matching method, OM法）

主要的分词方法

- 基于统计的机器学习方法
  - HMM、CRF、SVM、深度学习
- 机器学习算法和词典相结合，一方面能够提高分词准确率，另一方面能够改善领域适应性。
- 分词器遇到的几个问题
  - 不同分词器的分词标准经常不同
  - 歧义词问题
  - 新词问题

英文特征词

- 一般采用keyword,无需分词,单词之间有空格分开。
- 停用词（stop word），指文档中出现的连词，介词，冠词等并无太大意义的词。例如在英文中常用的停用词有the，a, it等；在中文中常见的有“是”，“的”，“地”等。
- 索引词（标引词，关键祠）:可以用于指代文档内容的预选词语,一般为名词或名词词组。
- 词干提取
  - countries => country，interesting => interest

权重计算方法

- 布尔权重(boolean weighting)
  - a_{ij}=1(TF_{ij}>0) or 0(TF_{ij}=0)
- TFIDF型权重
  - TF: $a_{ij}=TF_{ij}$
  - **TF*IDF: $a_{ij}=TF_{ij}*\log(N/DF_i)$**
  - TFC: 对上面进行归一化
  - LTC: 降低TF的作用
- 基于熵概念的权重(Entropy weighting)
  - 称为term i的某种熵
  - 如果term分布极度均匀：熵等于-1
  - 只在一个文档中出现：熵等于0

$$a_{ij}=\frac{TF_{ij}*\log(N/DF_i)}{\sqrt{\sum_k[TF_{kj}*\log(N/DF_k)]^2}}$$

$$a_{ij}=\frac{\log(TF_{ij}+1.0)*\log(N/DF_i)}{\sqrt{\sum_k[\log(TF_{kj}+1.0)*\log(N/DF_k)]^2}}$$

$$a_{ij}=\log(TF_{ij}+1.0)*\left(1+\frac{1}{\log N}\sum_{j=1}^{N}[\frac{TF_{ij}}{DF_i}\log(\frac{TF_{ij}}{DF_i})]\right)$$

目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的常用方法
    - 主题模型
    - word2vec

特征选择

- 基于DF
  - Term的DF小于某个阈值去掉(太少，没有代表性)
  - Term的DF大于某个阈值也去掉(太多，没有区分度)
- 信息增益(Information Gain, IG)：该term为整个分类所能提供的信息量(不考虑任何特征的熵和考虑该特征后的熵的差值)

$$\begin{aligned}\text{Gain(t)} &= \text{Entropy}\ (S) - \text{Expected}\ \ \text{Entropy}(\ S_t) \\ &= \{-\sum_{i=1}^{M} P(c_i)\log P(c_i)\} - \\ &\quad [P(t)\{-\sum_{i=1}^{M} P(c_i \mid t)\log P(c_i \mid t)\} + \\ &\quad P(\bar{t})\{-\sum_{i=1}^{M} P(c_i \mid \bar{t})\log P(c_i \mid \bar{t})\}]\end{aligned}$$

特征选择

- term的熵：该值越大，说明分布越均匀，越有可能出现在较多的类别中；该值越小，说明分布越倾斜，词可能出现在较少的类别中

$$Entropy(t) = -\sum_i P(c_i \mid t) \log P(c_i \mid t)$$

- 相对熵(not 交叉熵)：也称为KL距离(Kullback-Leibler divergence)，反映了文本类别的概率分布和在出现了某个特定词汇**(t)**条件下的文本类别的概率分布之间的距离，该值越大，词对文本类别分布的影响也大。

$$CE(t) = \sum_i P(c_i \mid t) \log \frac{P(c_i \mid t)}{P(c_i)}$$

特征选择

- χ^2 统计量：度量两者(term和类别)独立性的缺乏程度， χ^2 越大，独立性越小，相关性越大(若AD<BC,则类和词独立, N=A+B+C+D)

$$\chi^2(t,c)=\frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)}$$

	c	~c
t	*A*	*B*
~t	*C*	*D*

$$\chi^2_{AVG}(t)=\sum_{i=1}^{m}P(c_i)\chi^2(t,c_i)$$

$$\chi^2_{MAX}(t)=\max_{i=1}^{m}\{\chi^2(t,c_i)\}$$

互信息(Mutual Information)：MI越大t和c共现程度越大

$$I(t,c)=\log\frac{P(t\wedge c)}{P(t)P(c)}=\log\frac{P(t|c)}{P(t)}=\log\frac{A\times N}{(A+C)(A+B)}$$

$$I_{AVG}(t)=\sum_{i=1}^{m}P(c_i)I(t,c_i)$$

$$I_{MAX}(t)=\max_{i=1}^{m}P(c_i)I(t,c_i)$$

特征选择

- Robertson & Sparck Jones公式

$$RSJ\ (t, c_j) = \frac{c_j\text{中出现}\ \ t\text{的概率}}{\text{非}\ c_j\text{中出现}\ \ t\text{的概率}} = \log \frac{P(t \mid c_j)}{P(t \mid \overline{c}_j)}$$

$$TSV\ (t, c_j) = r * \log \frac{P(t \mid c_j)}{P(t \mid \overline{c}_j)}, r\text{为出现}\ \ t\text{的}\ c_j\text{类文档个数}$$

- 其他
  - Odds:

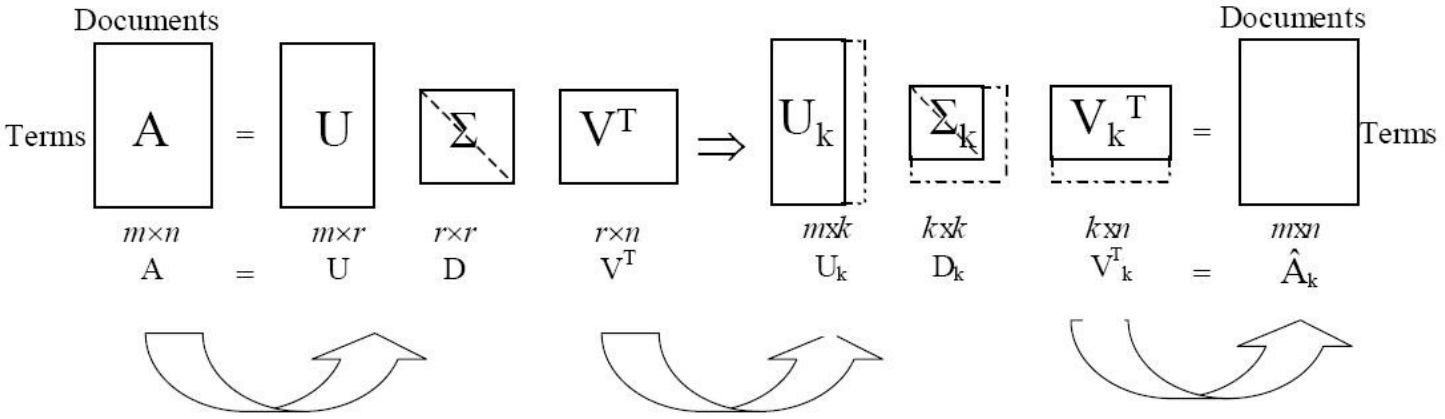
$$\frac{\log\ \ P\ (t \mid c_j)\ \log(\ \ 1\ -\ \ P\ (t \mid \overline{c}_j))}{\log(\ \ 1\ -\ \ P\ (t \mid c_j))\ \ \log\ \ \ P\ (t \mid \overline{c}_j)}$$

  - Term Strength:

$$P\ (t \in\ \ y \mid t \in\ \ x),\ x,\ y\text{是相关的两篇文档}$$

特征重构

- 隐性语义索引(LSI)
  - 奇异值分解(SVD)：
  - $A=(a_{ij})=UDV^T$
    - A_{M*N}, U_{M*r}, D_{r*r}(对角阵), V_{N*r}, r<=MIN(M,N)
  - 取D对角上的前k个元素(奇异值)，得D_k
    - $A_k=U_kD_kV_k^T$, U_k由U的前k列组成，V_k由V的前k列组成
    - 文档d在LSI对应的向量$d'=d^TU_kD^{-1}$ （$\mathbf{D^{1/2}V^T_k}$）的第d列



SVD

- $A=(a_{ij})=UDV^T$

A	d1	d2	d3	d4	d5	d6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	0	0	0	1	1	0
tree	0	0	0	1	0	1

SVD

- $A=(a_{ij})=UDV^T$

A

	d1	d2	d3	d4	d5	d6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	0	0	0	1	1	0
tree	0	0	0	1	0	1

U	1	2	3	4	5
ship	0.591	0	-0.737	0	0.328
boat	0.328	0	0.591	0	0.737
ocean	0.737	0	0.328	0	-0.591
wood	0	0.707	0	-0.707	0
tree	0	0.707	0	0.707	0

D

	1	2	3	4	5
1	1.801	0	0	0	0
2	0	1.732	0	0	0
3	0	0	1.247	0	0
4	0	0	0	1.00	0
5	0	0	0	0	0.445

V^T	d1	d2	d3	d4	d5	d6
1	0.737	0.591	0.328	0	0	0
2	0	0	0	0.816	0.408	0.408
3	0.328	0.737	0.591	0	0	0
4	0	0	0	0	0.707	0.707
5	0.591	0.328	0.737	0	0	0
6	0	0	0	0.577	0.577	0.577

SVD

- $A=(a_{ij})=UDV^T$

A	d1	d2	d3	d4	d5	d6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	0	0	0	1	1	0
tree	0	0	0	1	0	1

U_3	1	2	3
ship	0.591	0	-0.737
boat	0.328	0	0.591
ocean	0.737	0	0.328
wood	0	0.707	0
tree	0	0.707	0

D_3	1	2	3
1	1.801	0	0
2	0	1.732	0
3	0	0	1.247

V_3^T	d1	d2	d3	d4	d5	d6
1	0.737	0.591	0.328	0	0	0
2	0	0	0	0.816	0.408	0.408
3	0.328	0.737	0.591	0	0	0

SVD

A	d1	d2	d3	d4	d5	d6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
wood	0	0	0	1	1	0
tree	0	0	0	1	0	1

A’	d1	d2	d3	d4	d5	d6
ship	1.086	-0.048	0.892	0	0	0
boat	0.194	0.892	-0.242	0	0	0
ocean	0.845	1.086	0.194	0	0	0
wood	0	0	0	1	0.5	0
tree	0	0	0	1	0.5	01

http://blog.csdn.net/m0_37788308/article/details/78115313?locationNum=5&fps=1

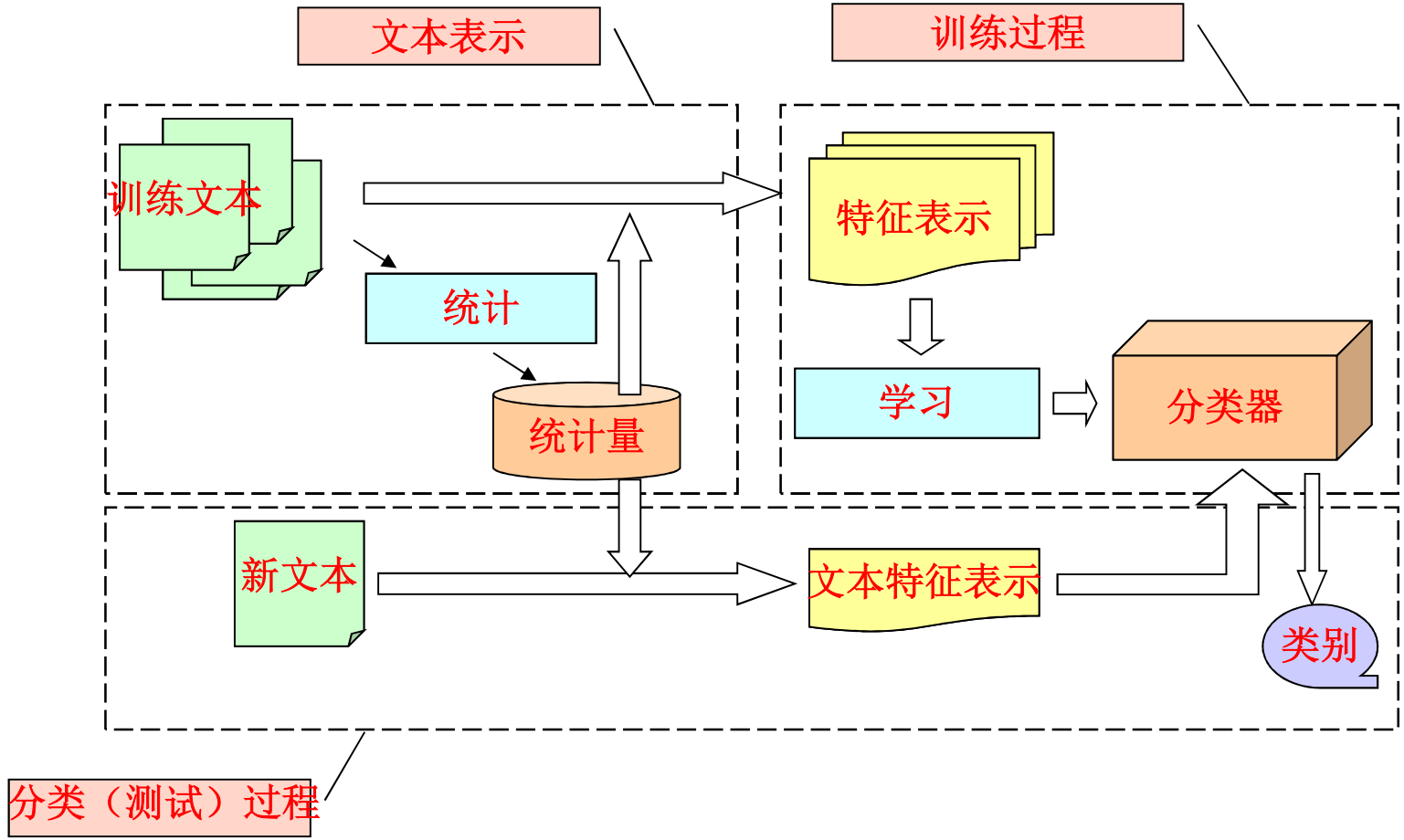
目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的常用方法
    - 主题模型
    - word2vec

文本分类

- 定义：给定分类体系，将文本分到某个或者某几个类别中。
  - 分类体系一般人工构造
    - 政治、体育、军事
    - 中美关系、恐怖事件
  - 分类系统可以是层次结构，如yahoo!
  - 分类模式
    - 2类问题，属于或不属于(binary)
    - 多类问题，多个类别(multi-class)，可拆分成2类问题
    - 一个文本可以属于多类(multi-label)
  - 这里讲的分类主要基于内容
  - 很多分类体系: Reuters分类体系、中图分类

文本分类的过程



自动文本分类方法

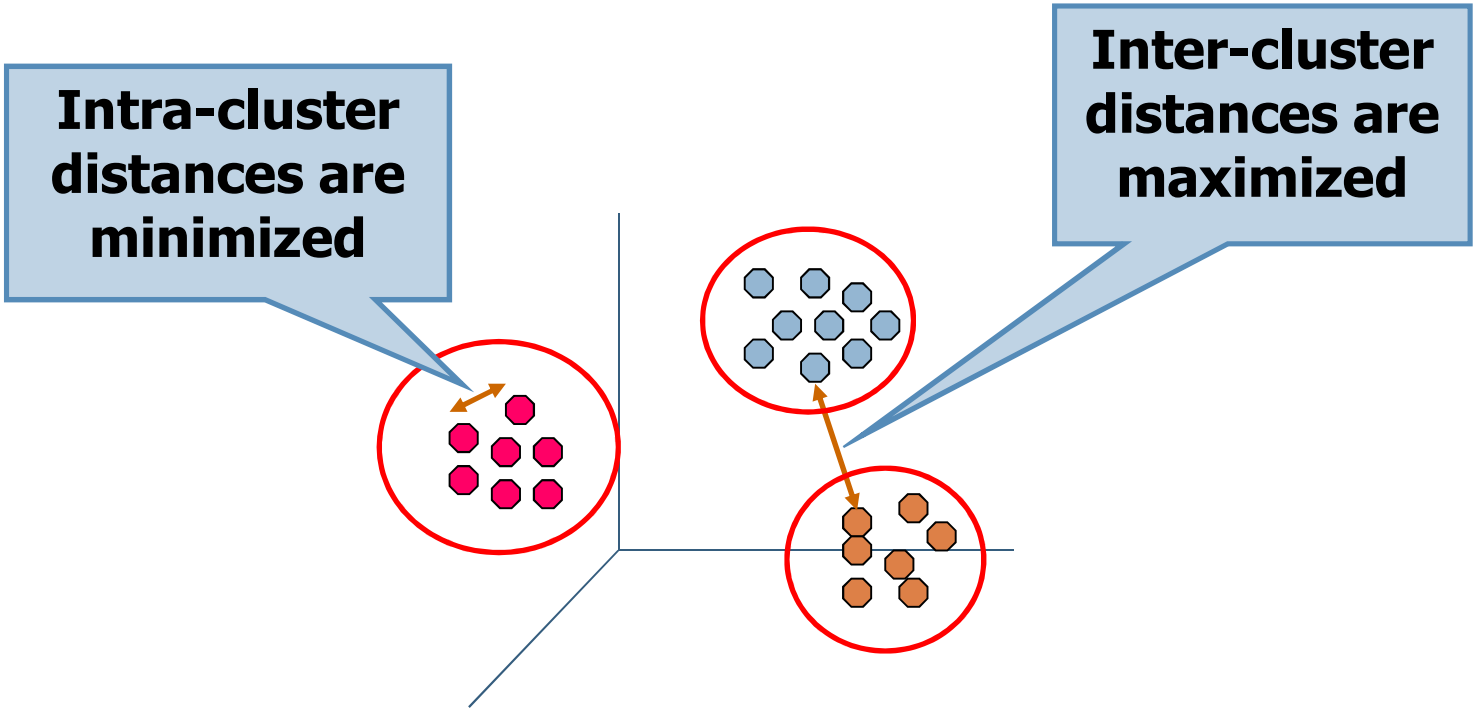
- Rocchio方法
- Naïve Bayes
- kNN方法
- 决策树方法decision tree
- Decision Rule Classifier
- The Widrow-Hoff Classifier
- 神经网络方法Neural Networks
- 支持向量机SVM
- 基于投票的方法(voting method)

目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的常用方法
    - 主题模型
    - word2vec

文本聚类

- 文本聚类是根据文本数据的不同特征，将其划分为不同数据簇的过程
- 其目的是要使同一类别的文本间的距离尽可能小，而不同类别的文本间的距离尽可能的大



文本聚类

- 文档自动聚类的步骤

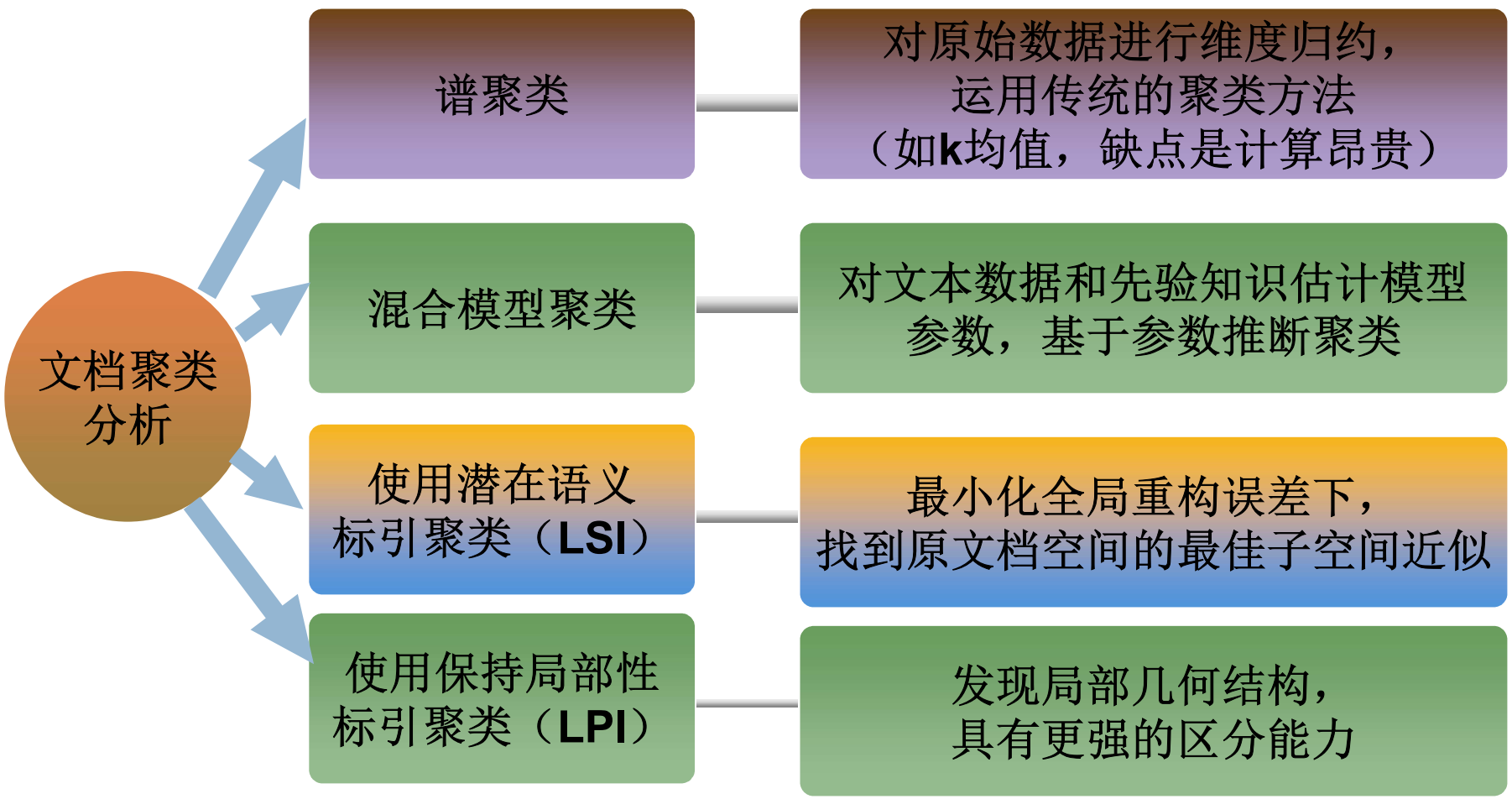
（1）获取结构化的文本集

（2）执行聚类算法，获得聚类谱系图(层次聚类时使用)。聚类算法的目的是获取能够反映特征空间样本点之间的“抱团”性质

（3）在得到聚类谱系图后，领域专家凭借经验，并结合具体的应用场合确定阈值

（4）执行聚类算法，获得聚类结果

文本聚类


文档聚类分析
谱聚类
对原始数据进行维度归约，运用传统的聚类方法（如k均值，缺点是计算昂贵）
混合模型聚类
对文本数据和先验知识估计模型参数，基于参数推断聚类
使用潜在语义标引聚类（LSI）
最小化全局重构误差下，找到原文档空间的最佳子空间近似
使用保持局部性标引聚类（LPI）
发现局部几何结构，具有更强的区分能力

文本聚类

- 文档自动聚类的类型
  - **平面划分法**：对包含n个样本的样本集构造样本集的k个划分，每个划分表示一个聚簇
  - **层次聚类法**：层次聚类法对给定的样本集进行层次分解。根据层次分解方向的不同可分为凝聚层次聚类和分裂层次聚类
  - **基于密度的方法**：根据样本点临近区域的密度进行聚类，使在给定区域内至少包含一定数据的样本点
  - **基于网格的方法**：采用多分辨率的网格数据结构，将样本空间量化为数量有限的网格单元，所有聚类操作都在网格上进行
  - **基于模型的方法**：为每个簇假定一个模型，然后通过寻找样本对给定模型的最佳拟合进行聚类

文本聚类

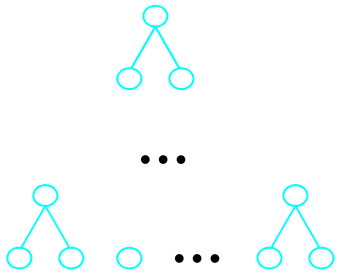
- 平面划分法
- 将文档集D=$\{d_1, \ldots, d_i, \ldots, d_n\}$水平地分割为的若干类，具体过程：

1. 确定要生成的类的数目 k；
2. 按照某种原则生成k个聚类中心作为聚类的种子 S=$\{s_1, \ldots, s_j, \ldots, s_k\}$；
3. 对D中的每一个文档d_i，依次计算它与各个种子s_j的相似度sim(d_i, s_j)；
4. 选取具有最大的相似度的种子arg max sim(d_i, s_j), 将d_i归入以s_j为聚类中心的类C_j，从而得到D的一个聚类C=$\{c_1, \ldots, c_k\}$；
5. 重复步骤2~4若干次，以得到较为稳定的聚类结果。

该方法速度快，但k要预先确定，种子选取难

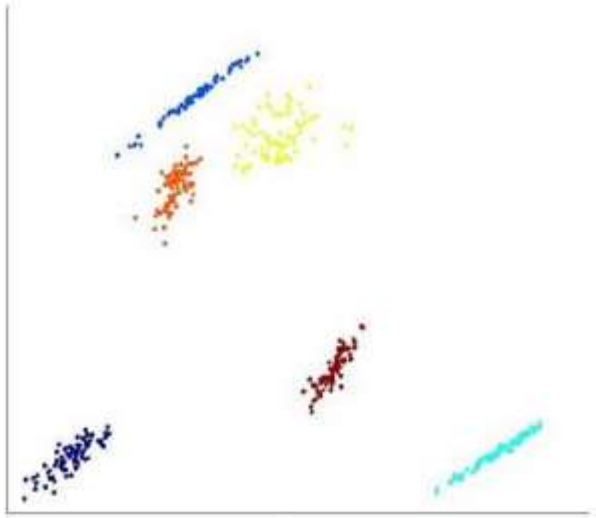
文本聚类

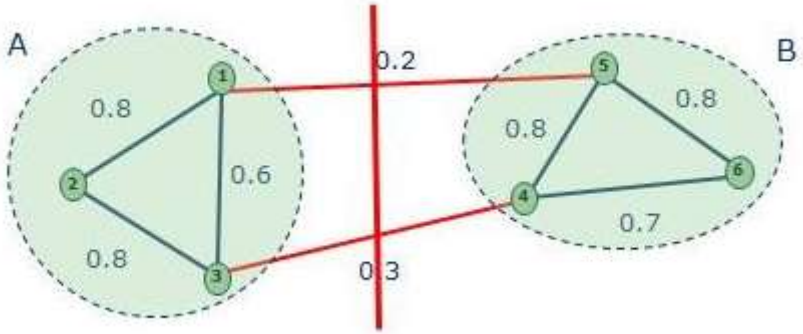
- 层次聚类法
- 具体过程
  - 将文档集D=$\{d_1, \ldots, d_i, \ldots, d_n\}$ 中的每一个文档d_i看作是一个具有单个成员的类$C_i=\{d_i\}$，这些类构成了D的一个聚类$C=\{c_1, \ldots, c_i, \ldots, c_n\}$；
  - 计算C中每对类（ c_i, c_j ）之间的相似度$sim(c_i, c_j)$；
  - 选取具有最大相似度的类对$\arg\max sim(c_i, c_j)$,并将c_i和c_j合并为一个新的类$c_k=c_i \cup c_j$，从而构成D的一个新的类$C=\{c_1, \ldots, c_{n-1}\}$；
  - 重复上述步骤，直到C中只剩下一个类为止。



谱聚类

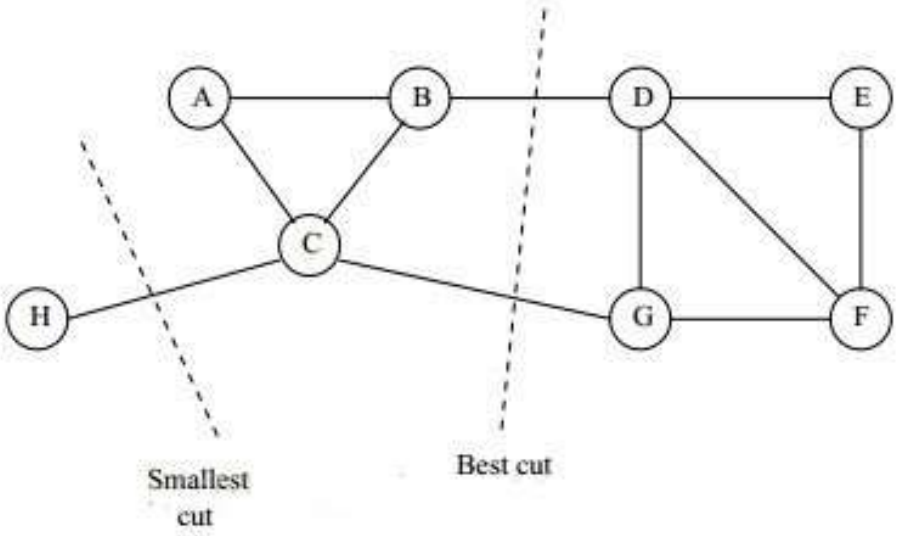
- 与传统的聚类算法相比，谱聚类具有能在任意形状的样本空间上聚类且收敛于全局最优解的优点。





谱聚类

- 谱聚类算法建立在图论中的谱图理论基础上，其本质是将聚类问题转化为图的最优划分问题
- 谱聚类的思想是将样本看作图上的顶点，样本间的相似度看作带权的边，然后将这个图分割成 k 个子图，并且使连接这些子图的边的权重和尽可能的低。
- 谱聚类能达到降维的作用，更适用于稀疏高维的文本数据



谱聚类

- 两个顶点的边要怎样定义？
  - 如果两个点在一定程度上相似，就在两个点之间添加一条边。顶点间相似度的计算可以采用k-means算法常用的相似度计算方法。谱聚类通常可以采用高斯相似方程来度量顶点间的相似度：

$$s(x_i, x_j) = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}}$$

- 要筛选掉哪些边？
  - 由于全连通图边太多不好处理，而且权重太低的边意味着这两个点相似度极地，几乎不可能属于同一个类别，因此可以筛选掉。谱聚类通常用的筛选边的方法是建立k-nearest 图，即仅保留每个顶点相连边中的k 个相似度最高的边，其余的筛选掉。

谱聚类

- Laplacian矩阵的定义
- 也称为基尔霍夫矩阵，是图的一种矩阵表示形式，拥有很多优秀的性质，适合研究图的特征
- 对于一个有个顶点的图 G(V,E)，其Laplacian矩阵定义为：
- $$L = D - W$$
- 其中，D为图的度矩阵，W为图的邻接矩阵

$$D = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & d_n \end{pmatrix} \qquad W = \begin{pmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & & \vdots \\ w_{n1} & \cdots & w_{nn} \end{pmatrix} \qquad d_i = \sum_{j=1}^{n} w_{ij}$$

谱聚类

- 拉普拉斯矩阵L 的性质：
- 对于任何向量 $f \in \mathbf{R}^n$:

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

- 所以，L 是对称矩阵和半正定矩阵。
- L 的唯一最小特征值是0，其对应的特征向量是**1**，即各个元素都为1 的向量。
  - L 有n 个非负的实数特征值 $0 = \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$

谱聚类

- 谱聚类算法将样本聚类问题转化成图分割问题。
- 假设W(S,T) 表示的是类别S 和T 之间所有边的权重之和, $\overline{A}$ 是A的补集。
- 对于k 个不同的类别 $A_1, A_2, \cdots, A_k$ ， 聚类问题就转化为最小化下面的目标函数Cut：

$$Cut(A_1, A_2, \cdots, A_k) = \frac{1}{2}\sum_{i=1}^{k} W(A_i, \bar{A}_i)$$

- 最小化这个目标函数，很容易出现孤立顶点被分割出来的情况，更希望每个类别的数量尽量均匀。

谱聚类

- 将聚类问题转化为最小化RatioCut问题:

$$RatioCut(A_1, A_2, \cdots, A_k) = \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^{k}\frac{Cut(A_i, \bar{A}_i)}{|A_i|}$$

- 尽管形式简单，但最小化RatioCut 却是一个NP 难问题，不方便求解。
- 为了找到解决办法，需要对目标函数进行转换。
- 对于二个类别的聚类问题，优化的目标函数为：

$$min \quad RatioCut(A, \bar{A}) = \frac{Cut(A, \bar{A})}{|A|} + \frac{Cut(A, \bar{A})}{|\bar{A}|}$$

谱聚类

- 令V 表示图中的所有顶点的集合，首先定义一个N 维向量f:

$$f_i = \begin{cases} \sqrt{|\bar{A}|/|A|}, & if \quad v_i \in A, \\ -\sqrt{|A|/|\bar{A}|}, & if \quad v_i \in \bar{A}. \end{cases}$$

- 利用之前提到的拉普拉斯的性质可得：

$$\begin{aligned} f'Lf &= \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \\ &= \sum_{i\in A, j\in \bar{A}} w_{ij}\left(\sqrt{\frac{|\bar{A}|}{|A|}} + \sqrt{\frac{|A|}{|\bar{A}|}}\right)^2 + \sum_{i\in \bar{A}, j\in A} w_{ij}\left(-\sqrt{\frac{|\bar{A}|}{|A|}} - \sqrt{\frac{|A|}{|\bar{A}|}}\right)^2 \\ &= Cut(A,\bar{A})\left(\frac{|\bar{A}|}{|A|} + \frac{|A|}{|\bar{A}|} + 2\right) \\ &= Cut(A,\bar{A})\left(\frac{|\bar{A}|+|A|}{|A|} + \frac{|A|+|\bar{A}|}{|\bar{A}|}\right) \\ &= |V| \cdot RatioCut(A,\bar{A}) \end{aligned}$$

谱聚类

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 = |V| \cdot RatioCut(A, \bar{A})$$

- 其中，|V| 表示的是顶点的数目，对于一个确定的图来说是个常数。
- 因此, 最小化*RatioCut* 就等价于最小化 $f'Lf$

$$min \quad f'Lf$$

$$s.t. \quad f \in R^n, \quad f \perp \mathbf{1}, \quad \|f\| = \sqrt{n}$$

- 其中

$$f_i = \begin{cases} \sqrt{|\bar{A}|/|A|}, & if \quad v_i \in A, \\ -\sqrt{|A|/|\bar{A}|}, & if \quad v_i \in \bar{A}. \end{cases}$$

谱聚类

- 因此，最小化RatioCut 值转化为求得L 的最小特征值

$$Lf = \lambda f$$

$$\Rightarrow f'Lf = \lambda f'f$$

$$\Rightarrow f'Lf = \lambda n$$

- 因此，最小化RatioCut 值转化为求得L 的最小特征值。
  - k分类即是求最小的k个特征值

谱聚类的计算量一般在于求解特征值这一部分，由于现在有一些快速求解或估计特征值的方法，所以计算量相对变得较小

谱聚类

- 由拉普拉斯矩阵的性质可知，拉普拉斯矩阵的最小特征值为0, 对应的特征向量不满足 $f \perp \mathbf{1}$ ，因此取第2小的特征值。
- 对于求解出来的特征向量 $f = (f_1, f_2, \cdots, f_n)' \in R^n$ 中的每一个分量 f_i， 根据每个分量的值判断对应样本点的分类：

$$\begin{cases} v_i \in A, & if \quad f_i \geqslant 0, \\ v_i \in \bar{A}, & if \quad f_i < 0. \end{cases}$$

- 也可以用k=2的k-means将特征向量 f 的元素分为两类

谱聚类

- 若对应的 k 个特征向量为 $f^{(1)}, f^{(2)}, \cdots, f^{(k)}$ ，这样便由特征向量构成如下的特征向量矩阵：

$$\begin{matrix} n \\ \end{matrix} \begin{pmatrix} f_1^{(1)} & f_1^{(2)} & \cdots & f_1^{(k)} \\ f_2^{(1)} & f_2^{(2)} & \cdots & f_2^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ f_n^{(1)} & f_n^{(2)} & \cdots & f_n^{(k)} \end{pmatrix}$$

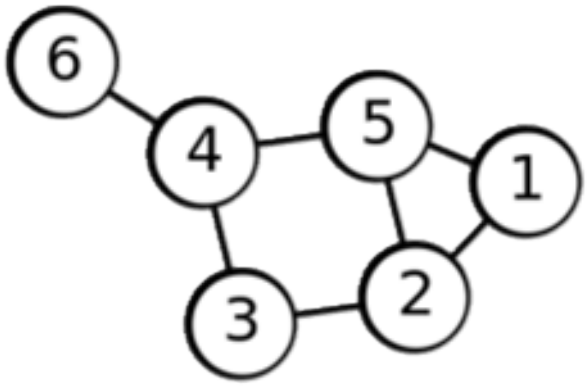
k

- 将特征向量矩阵中的每一行作为一个样本点，利用K-means聚类方法对其进行聚类，得到每个样本点所属的类别

相当于把每个样本的维度降为k

谱聚类算法步骤

- 步骤1：
- 将需要聚类的样本构建成一个图，图的每一个节点对应一个样本，将相似的点连接起来，并且用样本间的相似度作为两点间边的权重。将这个图用邻接矩阵表示，记为矩阵W。



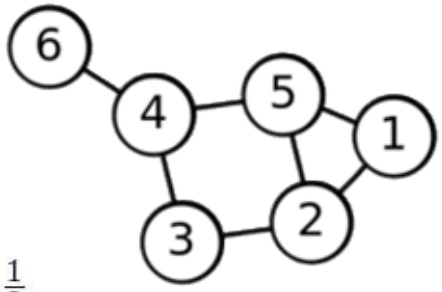
$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

邻接矩阵W

谱聚类算法步骤

步骤2：

- 计算这个图的度矩阵，记为D。
- 计算这个图的拉普拉斯矩阵 $L = D - W$
- 规范化的拉普斯矩阵 $L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$



$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

拉普拉斯矩阵L **度矩阵D** **邻接矩阵W**

谱聚类算法步骤

- 步骤3：
  - 求出矩阵L 的k 个最小的特征值 $\lambda_{i=1}^{k}$ 以及对应的特征向量 $f_{i=1}^{k}$
- 步骤4：
  - 将这k 个特征列向量排列在一起组成一个$N*k$ 的矩阵，其中每一行看作k 维空间中的一个向量
  - 使用K-means 算法对这个$N*k$ 的矩阵进行聚类。
  - 聚类的结果中每一行所属的类别就是原来图中的顶点所属的类别，得到每个样本所属的类别。

$$\mathbf{N}\begin{pmatrix} f_1^{(1)} & f_1^{(2)} & \cdots & f_1^{(k)} \\ f_2^{(1)} & f_2^{(2)} & \cdots & f_2^{(k)} \\ \vdots & \vdots & \ddots & \vdots \\ f_n^{(1)} & f_n^{(2)} & \cdots & f_n^{(k)} \end{pmatrix}$$

k

谱聚类的优点

- 与传统的聚类算法K-means相比：
- ① 复杂度更低
  - K-means的时间复杂度为O(TKMN)，T为迭代次数，K为类别数，M为样本数，N为样本维度
  - 当样本维数增大时，K-means的计算会困难
  - 谱聚类通过降维的方式，将样本的维度降为K<<N
- ② 对数据要求低
  - K-means 要求数据必须是 N 维欧氏空间中的向量
  - 谱聚类只需要数据之间的相似度矩阵

谱聚类的优点

- ③ 更健壮
  - 谱聚类由于抓住了主要矛盾，忽略了次要的东西，因此比传统的聚类算法更加健壮一些，对于不规则的误差数据不是那么敏感，而且聚类结果也要好一些。
- ④ 对于稀疏矩阵更有效，例如文本数据
  - 虽然最初的数据是稀疏矩阵，但是 K-means 中有一个求 Centroid 的运算，就是求一个平均值：许多稀疏的向量的平均值求出来并不一定还是稀疏向量。
  - 这时再计算向量之间的距离的时候，运算量就变得非常大。
  - 谱聚类对稀疏矩阵求特征值和特征向量有很高效的办法，得到的结果是一些 k 维的向量（通常 k 不会很大），在这些低维的数据上做 K-means 运算量非常小

谱聚类

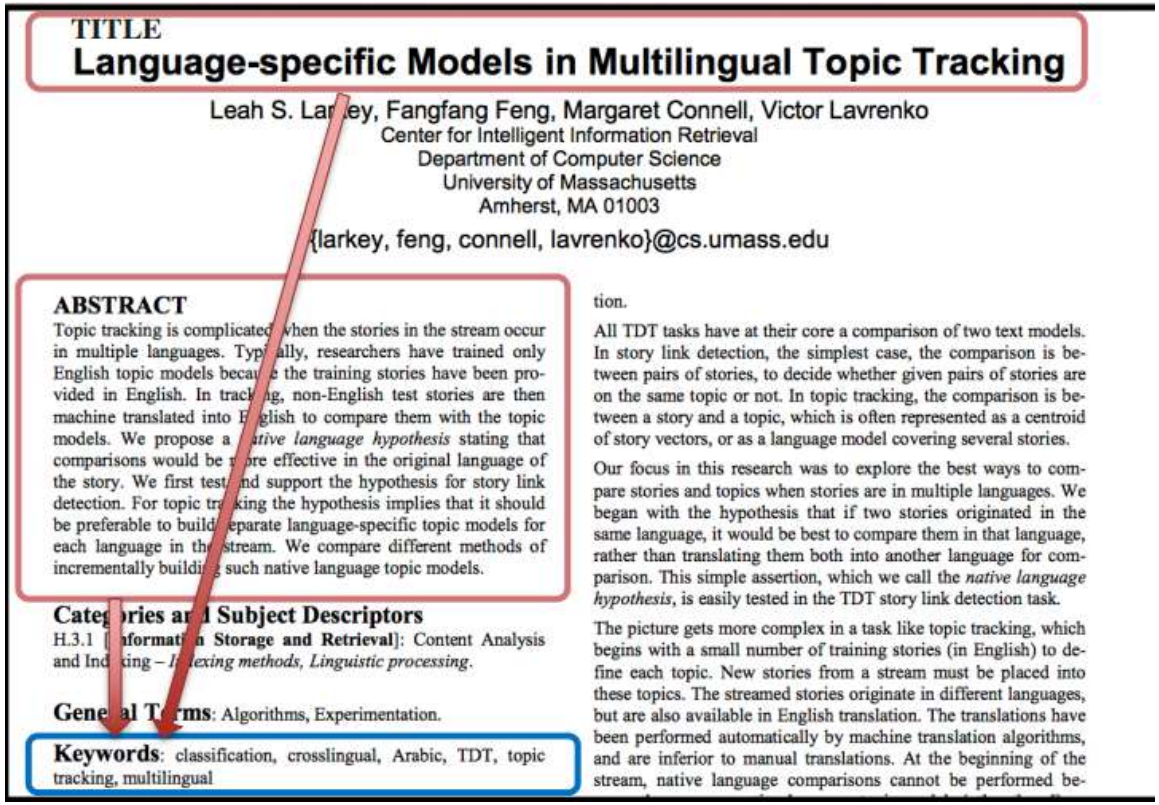
- 利用设置超级点减低大规模数据谱聚类的复杂度：
  - Liu J, Wang C, Danilevsky M, et al. Large-scale spectral clustering on graphs[C]//Proceedings of the Twenty-Third international joint conference on Artificial Intelligence. AAAI Press, 2013: 1486-1492.
- 增量聚类
  - Ning H, Xu W, Chi Y, et al. Incremental spectral clustering by efficiently updating the eigen-system[J]. Pattern Recognition, 2010, 43(1): 113-127.
- 谱聚类的半监督学习
  - Chen W, Feng G. Spectral clustering: a semi-supervised approach[J]. Neurocomputing, 2012, 77(1): 229-242.
- 分布式并行下的谱聚类
  - Chen W Y, Song Y, Bai H, et al. Parallel spectral clustering in distributed systems[J]. IEEE transactions on pattern analysis and machine intelligence, 2011, 33(3): 568-586.

目录

- 传统的文本处理
  - 自然语言处理
  - N-Gram语言模型
- 文本挖掘
  - 文本挖掘简介
  - 特征抽取
  - 特征选择
  - 文本分类
  - 文本聚类
  - 文本挖掘的应用实例
    - Keyphrase Extraction

Keyphrase Extraction Methods

- 关键词提取是提取能高度概括文章内容的一组短语
- 输入：一个文档，输出：一些短语集合
- 评价指标：和人工分配的关键短语比较，计算准确率，召回率，F值



TITLE

Language-specific Models in Multilingual Topic Tracking

Leah S. Larkey, Fangfang Feng, Margaret Connell, Victor Lavrenko
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{larkey, feng, connell, lavrenko}@cs.umass.edu

ABSTRACT

Topic tracking is complicated when the stories in the stream occur in multiple languages. Typically, researchers have trained only English topic models because the training stories have been provided in English. In tracking, non-English test stories are then machine translated into English to compare them with the topic models. We propose a *native language hypothesis* stating that comparisons would be more effective in the original language of the story. We first test and support the hypothesis for story link detection. For topic tracking the hypothesis implies that it should be preferable to build separate language-specific topic models for each language in the stream. We compare different methods of incrementally building such native language topic models.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *Indexing methods, Linguistic processing.*

General Terms: Algorithms, Experimentation.

Keywords: classification, crosslingual, Arabic, TDT, topic tracking, multilingual

tion.

All TDT tasks have at their core a comparison of two text models. In story link detection, the simplest case, the comparison is between pairs of stories, to decide whether given pairs of stories are on the same topic or not. In topic tracking, the comparison is between a story and a topic, which is often represented as a centroid of story vectors, or as a language model covering several stories.

Our focus in this research was to explore the best ways to compare stories and topics when stories are in multiple languages. We began with the hypothesis that if two stories originated in the same language, it would be best to compare them in that language, rather than translating them both into another language for comparison. This simple assertion, which we call the *native language hypothesis*, is easily tested in the TDT story link detection task.

The picture gets more complex in a task like topic tracking, which begins with a small number of training stories (in English) to define each topic. New stories from a stream must be placed into these topics. The streamed stories originate in different languages, but are also available in English translation. The translations have been performed automatically by machine translation algorithms, and are inferior to manual translations. At the beginning of the stream, native language comparisons cannot be performed be-

Keyphrase Extraction

- 自动提取关键词主要分为两步
  - 用启发式规则选取一些短语作为候选短语
    - 名词短语，模式为(adjective)* (noun)+ **(0个或多个形容词后接着1个或多个名词)**
      - Probabilistic algorithms, relational probabilistic model, summarization
    - 短语(不含有提前定义好的停用词)
  - 使用有监督（分类）或无监督（聚类）的方法从这些候选短语中选择关键短语

Keyphrase Extraction Methods

- TF-IDF

TF-IDF考虑两个因素单词在文档d中的频率，以及多少个文档包含d,t是单词如下：

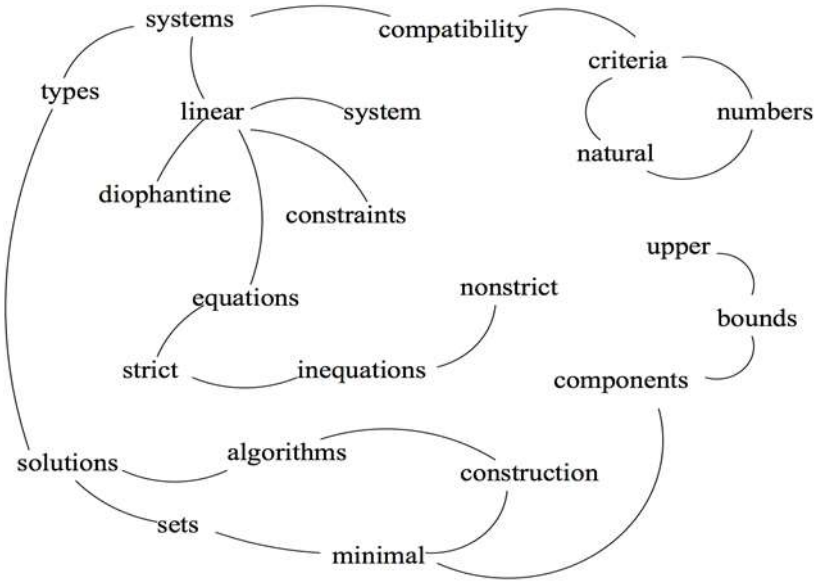
$$\text{tfidf}_t = tf_t \times \log(\frac{D}{D_t})$$

- tf_t：t在文档d中的频率
- D_t：含有t的文档数
- D：语料库中的文档总数

- 给定一个文档我们首先计算文档中每项的Tf-Idf分数
- 计算候选短语的得分(短语中每个单词的得分和)
- 输出得分最高的K个短语作为关键词

Keyphrase Extraction Methods

- TextRank [1]
- 动机：一个单词是重要的如果和它相关的单词也是重要的



Keyphrase Extraction Methods

- TextRank
- 在单词级别构建图
  - 只保留文档中的名词和形容词,每个词对应文档中的一个节点。如果两个节点对应的单词在文档中一个大小为w的窗口内出现，两个节点连边(图是无向的,有向无向差别不大)
- 在图上使用随机游走算法(如pagerank)得到每个节点的得分
- 计算候选短语的得分(短语中单词得分之和)

Keyphrase Extraction Methods

- 朴素贝叶斯分类器 [2]
- 将关键词提取当成分类问题，正样本是关键词，负样本是非关键词
- 对于文档中的每个候选短语提取两个特征:Tf-Idf得分，短语距离文档开始的距离。使用这两个特征用朴素贝叶斯分类器进行分类

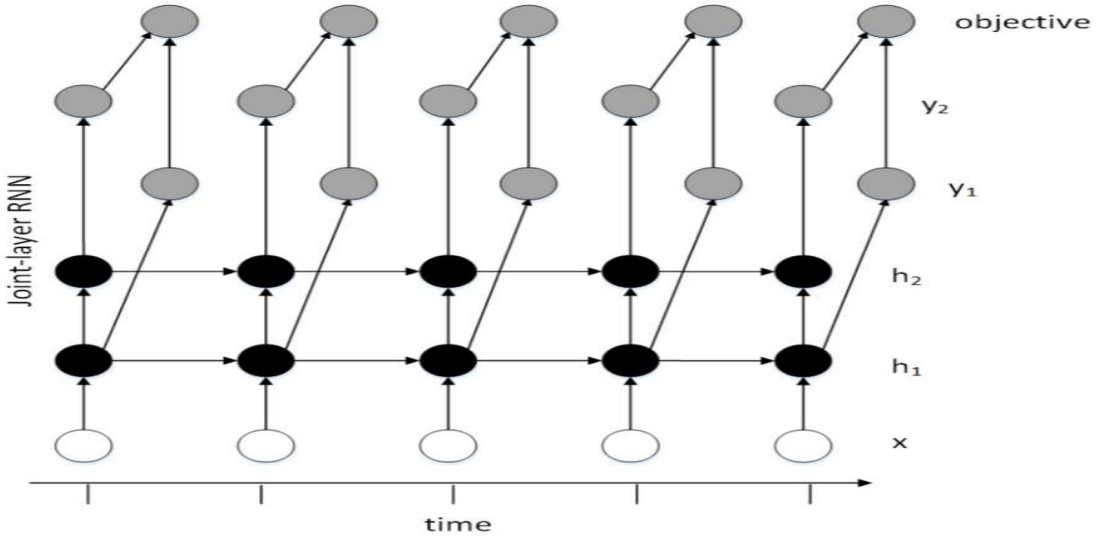
Keyphrase Extraction Methods

联合层循环神经网络[3]

- 由两层循环神经网络(RNN)构成
  - 第一层循环神经网络判断当前位置词是否是关键词（二分类）。
  - 第二层循环神经网络判断当前位置词是关键短语的哪一部分(开始，中间，结尾)或者单个词构成短语或者不是关键词（多分类）

第二层

第一层



References

[1] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pages 404–411.

[2] Carl Gutwin, Gordon Paynter, Ian Witten, Craig NevillManning, and Eibe Frank. 1999. Improving browsing in digital libraries with keyphrase indexes. Decision Support Systems, 27:81–104.

[3] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, “Keyphrase Extraction using Deep Recurrent Neural Networks on Twitter,” in EMNLP, 2016, pp. 836–845.

关键词抽取的一篇综述 Automatic Keyphrase Extraction: A Survey of the State of the Art