《统计自然语言处理基础》

图书基本信息

书名：《统计自然语言处理基础》

13位ISBN编号：9787505399211

10位ISBN编号：7505399217

出版时间：2005-1

出版社：电子工业出版社

作者：Chris Manning,Hinrich Sch ü tze

页数：418

译者：苑春法,李伟,李庆中

《统计自然语言处理基础》

内容概要

《统计自然语言处理基础：国外计算机科学教材系列》是一本全面系统地介绍统计自然语言处理技术的专著，被国内外许多所著名大学选为计算语言学相关课程的教材。《统计自然语言处理基础：国外计算机科学教材系列》涵盖的内容十分广泛，分为四个部分，共16章，包括了构建自然语言处理软件工具将用到的几乎所有理论和算法。全书的论述过程由浅入深，从数学基础到精确的理论算法，从简单的词法分析到复杂的语法分析，适合不同水平的读者群的需求。同时，《统计自然语言处理基础：国外计算机科学教材系列》将理论与实践紧密联系在一起，在介绍理论知识的基础上给出了自然语言处理技术的高层应用（如信息检索等）。在《统计自然语言处理基础：国外计算机科学教材系列》的配套网站上提供了许多相关资源和工具，便于读者结合书中习题，在实践中获得提高。近年来，自然语言处理中的统计学方法已经逐渐成为主流。

《统计自然语言处理基础》

书籍目录

《统计自然语言处理基础》

编辑推荐

《统计自然语言处理基础：国外计算机科学教材系列》不仅适合作为自然语言处理方向的研究生的教材，也非常适合作为自然语言处理相关领域的研究人员和技术人员的参考资料。

《统计自然语言处理基础》

精彩短评

1、并没有全部看完，有些地方还看不懂。但我知道它挺不错，特别是对于NLP的入门或进阶者
2、挺好的。。。显蓝没看完- - ps求过
3、其实我不懂统计学
4、经典
5、老书 翻翻
6、还是比较经典的，附带看看宗成庆那本
7、简短介绍，刚刚入门啊
8、看得稀里糊涂的。。
9、学习中。。。
10、导论
11、NLP不错的入门书籍
12、第一遍没全看懂，还是回炉一遍好了
13、经典之作，必读
14、看着玩吧 之前粗略的看过一遍
15、论文呐论文。
16、入门经典；书中有翻译错误
17、补充标注，这本书对写韩梅梅和了解一些nlp基础帮助很大，不过句法分析那里就不懂了，至少我不是靠看这本书懂的。后三章还是看看别的书比较好。
18、翻译太渣..以及..原来只学概率是不够的T_T
19、经典之作
20、只能说大概翻了一遍，真心把好几本书的统计和信息技术综合了，不错的书
21、翻译能再烂点么
22、NLP 读过的第一本书
23、读了前9章，涨涨基础。
24、经典入门书，就是比较老了。
25、超烂
26、翻译得还不错
27、:
H087/6432
28、对NLP中问题有了基本认识
29、自然语言处理必读经典书籍
30、上完了一学期的computational linguistic的课 就当我读过吧……

《统计自然语言处理基础》

精彩书评

1、P17（中文版）English：The significance of power laws中文：强法则的重要性power law：指数法则，幂律
2、还行，但比想象的要差。缺点：书翻译的很蹩脚，写得也有些蹩脚。书里充满了概念。一个特点：文字多的地方，基本感觉易读性比较差，说来说去不知道在说什么了。公式多的反而好理解一些。不该省略的地方省略了不少。比如2.1.10 贝叶斯，贝叶斯大学学过的，但是33页那个讲的是什么玩意。 还有那个什么噪声信号模型，一看到那么多的符号，根本就不想读了。计算条件熵时（P39），log(4/3)非要写成 2-log3，本来很简单的东西，吐血的浪费读者时间。到处都是文字，不知道哪儿是重点。HMM部分介绍，跟《模式识别》里的易读性不是一个级别。如果做中文文字处理，这本书会让人有些失望。基本都是分析英语。优点：书的优点还是很明显的。也内容比较权威。如果Aho出的那本《编译原理》是一流的书，这本书的（可读性上）只能算二流。
3、这本书不是很厚，也没有自然语言处理综论介绍的全面。但就想要学习SNLP的人来说相当不错。同时书中除了自然语言处理中传统的如分词、标注等领域之外，在最后也涉及到了一些较为新型和更为交叉的领域。从SNLP这一领域做出了很好的诠释！
4、power law译成强法则，perplexity译成混乱度，碰到稍难一点的句子居然直接跳过不译，狂汗。现在还没看多少，感觉原书内容还是不错的，叙述比较完备，就是英文写得稍微难了点，不是特别简单易懂的写法。

《统计自然语言处理基础》

章节试读

1、《统计自然语言处理基础》的笔记-第1页

第一本带了专业方向性的书,做点小记录
前半本有价值的东西不多..
另外..豆瓣上发东西要怎么才能让格式好看点...

#1.绪
=====
不关心是否合乎语法这一分类

语言的自我发展:
19c才出现kind of/sort of的程度修饰用法

起因:a kind of adj. n. -> (kind of adj.) n.

利用词性+语法的parse,歧义太多:
List the sales of the products produced in 1973 with the products produced in 1972
有455种句法分析结果(Martin)
增加限制与优选规则(只能手工添加)非常费力,且无法处理生动的(修辞)语言.
相反,统计方法自动归纳结构信息,挖掘搭配关系.

Garden Pathing现象:
The horse raced past the barn fell.
发现无法分析后要回溯到The horse.在口语中因为语气及停顿,不会有此问题

Brown语料库/Susanne语料库(free)

token- 总词次 type- 总词数

Zipf法则:大型语料库中,一个单词的词频与词频排名成反比.

Entropy: H(x) = -E[log(p(x))], H(X,Y) = -E[log(p(x,y))], H(Y|X) = -E[log (p(y|x))]

(chain rule:) H(X,Y) = H(X) + H(Y|X)

Mutual Information: I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)
已知一者对另一者不确定性的减少量
I(X;Y) = E[log(p(x,y) / p(x)p(y))]

噪声信道模型: 通过最大化编码的entropy, 可使得噪声信道的输入与输出的互信息最大化, 此时传输速率可等于信道capacity.

有限记忆性->k阶Markov链.
用一阶二阶模型估算英语的熵约为3.3-4, Shannon人工实验得到1.34

#4.2文本

========
文本程序预处理中的许多实际问题(格式与标点相关的识别困　难)

#6 N-gram model
==========
可以想象一个靠前k个词及谓语的预测会很可靠. 但谓语识别很困难.
trigram模型仍然是非常好的预测模型.
高阶n-gram模型只适合在巨量测试语料上使用.
如何给未登录词赋一个非0概率?
Laplace: 全部分子+1
Lidstone: +t, Jeffreys-perkes: t = 1/2时是期望似然估计
选一个小t以避免太多概率空间转移到未知事件.

6.2.3如何选取训练数据和测试数据
p138网站
利用n-gram的频率给出概率估计的各类技巧:
deleted estimation, deleted interpolation estimation, Good-Turing estimation

6.3:通过不同阶(较小)的n-gram频率来估计给定n值的n-gram概率,可以有助于数据稀疏问题
线性插值:P(wn|wn-2, wn-1) = a1P(wn) + a2P(wn|wn-1) + a3P(wn|wn-1, wn-2)
可利用EM算法(Expectation Maximization)确定最佳权值
Katz回退算法..
一般化线性插值: 系数权值是关于历史的函数

#7 语义消岐
==========
有监督学习:含语义标注数据. 无监督学习(clustering):聚类
以人类的标注成功率作为效果的上界,将所有词指定为其最常用语义作为其效果的下界

7.2有监督消岐
7.2.1 Bayes Classification
决定w的语义s:
s = max(s_k){P(s_k | c)} = max(s_k)P(c| s_k)P(s_k) (上下文c(context window)的概率为常量)
= max(s_k){log P(c| s_k) + log P(s_k) }
使用Naive Bayes Assumption, 赋予其独立性,则上式继续
= max(s_k){log P(s_k) + sum(v in c){log P(v | s_k) } }

其中的P(v|s), P(s)可从语料(必须是已标记好的)中利用MLE计算出(最好加上适当的平滑)
P(v_t|s) = C(v_t, s) / sum(v_i)C(v_i,s); P(s) = C(s) / C(words)

7.2.2 An Approach using Information Theroy
将w可能的语义集合P,与其对应指示器(如下文的两个单词)的可能集合Q做划分
P = {P1, P2, ...}, Q = {Q1, Q2, ...}
使得这样划分后互信息I(P, Q)最大,重复迭代二者可保证解的存在
w = {做} P = {take, make}, Q = {measure, note, decision}

7.3Dict-Based Disambiguation
7.3.1语义定义
判断w的语义时,挑选与w的附近词定义相似性最大的定义(看词语重复)

7.3.2类义
利用包含语义范畴信息的词典,每个词有若干个tag
在运行中可以对遇到的词添加新的tag(Classification)
类义的范畴有时与语义无法对应匹配,因此用于语义消岐效果不好.
7.3.3基于第二语言语料库. 利用目标词的上下文的翻译的上下文寻找结果.
7.3.4文本语义:一词在整篇文本中更可能取同一语义.

7.4无监督消岐
随机初始化P(v| s_k), 根据EM算法重新估计P(v| s_k), 使得模型整体的似然值保持增长

#8 词汇获取
==========
8.1 关于假设检验,拒真受伪
8.2 动词子范畴帮助句法分析
动词可引导什么样的结构. tell sb sth; find sb adj;
利用这样的范畴作出假设,检验,自我修复
8.3 附着歧义
一般使用简单的"同现计数"来统计即可
介词短语的附着歧义: 附着于动词还是名词?
一种自动学习方法: 先找出所有无歧义的
8.4 选择倾向
v/subject, v/object, adj/noun等搭配,在特定中心词下,常倾向于某一特定类型.
可对名词归类,与v和adj匹配
选择倾向过强的动词可能隐藏其宾语: He ate.
8.5 语义相似性
通用描述词汇的语义十分困难,因此词汇获取的最终结果往往落到语义相似性上
8.5.1 (名词)转换成文档空间(或修饰词空间)的向量(同现次数),评价相似度
向量相似性的度量: 除余弦外还有其他特征数. 余弦较常用
p203 计算余弦相似性的结果. 利用对数加权方程f(x)=1+log(x)代替同现计数

8.5.2 有时我们的数据是概率向量,而余弦基于欧氏距离,不适合评价概率向量的相似性.
概率分布中相似性的度量方法: 相对熵;信息半径;L1范式(Mahatton距离,即一阶幂平均)

8.6 词汇获取的重要性:
派生词等非辞典词的存在

#9 Markov
=========
n阶Markov模型,可视为一个状态有n元的一阶Markov模型.
1阶Markov模型就是一个正则概率转移图, 也可看作不确定有限状态自动机.

Hidden Markov Model: 过程未知
Viterbi Algorithm: 根据起止点找最可能路径. DP
HMM中对原始模型的参数估计:随机选取初始值,迭代修改,可得到局部最优值

实现中,有些主要依靠乘法的算法(如Viterbi)常用对数来实现, 更快速且减小了浮点误差

#10 POS tagging
==============

信息源(可用数据):
表. 不可靠: 几乎所有名词都可作为动词->对这类信息的高级描述得到dumb tagger
Markov tagger. 按照POS作为状态进行转移
句子w[]的最佳POS序列t[]为:
max(t){prod(i=1~n){P(w_i | t_i)* P(t_i | t_i-1)}}
仍然是SSP问题
对未登陆词的处理: 某些可根据词形猜测,一般的给每个未登陆词一个POS分布
三元tagger未必更好, 对二元/三元做插值也许会好. 阶数高时注意跨越标点时的处理方法

无初始数据的情形(对未知语言,无语料库; 或对特殊领域),用HMM tagger, 如何初始化模型参数是关键

Transformation-based Learning of Tags, 即学习-重写的过程
不容易出现过度适应测试集的情形..?

其他语言中,更多的词形变化可能能为POS tagging提供更多信息. 不同语言的标注集不可比,语法不同
一般语料的标注准确率已能达到95%+

Markov的弊病: 无法处理Recursive Gramma结构!
**将每个单词换成语法等价的多词短语,是否可破坏多数此类tagging算法?(Markov的limit horizon)
The velocity rises to -> The velocity of waves rises to. 难以处理,因为复数名词+单数动词少见

#11 Probabilistic Context Free Gramma

对POS建树后,利用转移概率, 计算这种结构的概率
e.g.: S-> NP +VP : 0.3; PP->P +NP : 1.0; NP -> stars : 0.18;
将各节点转移概率相乘即为树概率
ContextFree Hypothesis: 子树的概率与其他部分无关
PCFG的一个欠缺:未考虑词汇的同现特征,只基于结构.因此需要与上下文知识结合.
PCFG使得短句子的概率更大, 但事实上WallStreetJournal的句子平均长度为23
解决思路不是对结果再处理,而应是找到更好的(仍基于概率的)metrics of goodness
PCFG中所有合法句子的概率和,并不一定是1, 实际中没什么影响.

计算PCFG也可构造出子结构递推, 利用类似前向-后向的方法,有内部概率-外部概率
内部概率是此子树的概率:选择分割点,得到子结构
外部概率是除此子树外的部分的概率,自上而下计算.

记录内部概率做DP, 即可找句子的最佳句法分析结果

可用EM算法训练数据, 找到最大化语料库似然性的语法.

Problem: 复杂度/ 局部极值/ 叶节点个数/ 参数初始化

#12 Statistical Parsing

确定句子的树状结构及POS

PCFG缺乏词汇化,有些词汇转移生成P +NP的概率会更高;
在附着歧义中更是如此,仅仅有POS只能提供很少信息
PCFG的概率上下文无关假设非常错误..

Dependency Grammar: 考虑词汇之间(语义上的)依赖关系,摈弃了庞大的结构树形式

评价结果: PARSEVAL度量尺度 效果粗糙
准确率: 有多少个标准答案中的括号; 召回率:结果有多少个括号是标准答案;交叉括号.

一些看不懂的Parser..

#13 Machine Translation
==========
词的对应->词义消岐; 句法转换 ->句法分析消岐; 即使同句法,仍可能有语义歧义;
用语义级别的媒介, 又难以设计语义表达方法.

文本对齐: 所谓对齐,不允许交叉,而允许组块

Length-Based: 假设短句对应短句,长句对应长句. 同语系时很有效
不同语系中会有很多1:3,3:1,3:3模式.
假设只存在{1:1,1:0,0:1,1:2,2:1,2:2}这几种对齐方式,用f[i][j]表示前i+前j的匹配价值,做DP

Length-Based只能处理clean text. 若分割标记有噪声,无法识别句子边界就会悲剧.
考虑词汇间的对应关系,建立双语文本映射表.压缩为bitmap后寻找一条明显轨迹.

Word-Based: 基本假设:分布位置相同的词语是对应的
方法:选取首尾固定住,中间部分交叉分析,将分布接近的词语认为对应,固定住,重复上操作.
只考虑实词的匹配会更好.
词对齐的另一个效果是可以识别(翻译可能不同的)未登陆词

语言模型:得到句子a的生成概率;
翻译模型:句子a翻译成句子b的概率->对所有可能的对齐方式求和
<-需要知道词a翻译成词b的概率<-词语分布关联性
词语的一对多,多对一难以解决; 词态变化,短语,长距离语法结构都需靠语言知识学习;
好的模型应能够分析出句子成分间的对应关系,而不是通过一对多的数据输入.
好的模型不应有太多的独立性假设

#14 Clustering
==========
认为上下文信息足够提供词语的相似性
词相似: 利用上下文的模式相似度
Clustering算法不需要提供训练数据,无监督.
hierarchical vs. flat 类别间是否有层级. flat更为简洁高效
soft vs. hard 每个样本是否可属于多类,属于多个类的概率分布

Implement: 定义出类之间的相似度函数
bottom-up hierarchical algorithm: 每个对象都是一个类,不断合并最相似的两个类
top-down: 初始只有一个类,每次将内聚程度最小的一部分元素分出去. 要求相似函数对自变量单调递减性:并集相似度低

单连通clustering: 用两集合最相似样本的相似度衡量集合相似度. 容易产生chaining effect.
算法执行过程类似MST.由相似度单调性,可类似MST快速实现

全连通:用最不相似样本的相似度衡量相似度. 聚的更紧密. 复杂度更高(n^3)
平均连通: 也即利用余弦度量相似度,合并操作可高效完成. 复杂度为n^2

k-means算法(hard-clustering):任选k个中心,按距中心距离聚类,用各类元素均值更新中心.
EM算法:按距中心距离计算各元素属于各类的概率分布
广义EM算法的介绍:

#15 Information Retrieval
==========
Probability Ranking Principle:按照相关概率的降序排列文档(假设了文档间彼此不相关)
用n维向量空间衡量相似性. 文档在n个主题上的权重作为向量,对于normalized的向量,余弦与欧式距离给出的排序相同
若不进行normalized,则长文档的权重会高.

tf_{i,j}: w_i在d_j中出现的次数; df_i: 出现w_i的d的个数; cf_i: w_i出现的总次数
tfidf: w_{i,j} = (1 + log(tf_{i,j})) * log(D / df_i)
也有其他计算方案

IDF推导:
给定查询,我们需要按照odds of relevance: P(Rel | d) / P(NotRel | d)排序
对其用Bayes展开后取对数,得log P(d | Rel) - log P(d | NotRel) + log P(Rel) - log P(NotRel)
后两项与文档无关,排序时舍去
假设文档中的词独立出现,则P(d | Rel) = \prod_i{P(w_i出现 | Rel)}
上式对查询中的所有单词求积,取对数后,待排序函数变为:
O(d) = \sum_i(log P(X_i | Rel) - log P(X_i | NotRel)),其中X_i表示w_i在d中是否出现
设p_i = P(1 | Rel)表示w_i出现在相关文档中的概率,注意到P(X_i | Rel) = p_i^{X_i} * (1 - p_i)^{1-X_i}
同理定义q_i = P(1 | NotRel),函数变为:
O(d) = \sum_i { X_i * [log p_i/(1-p_i) + log (1-q_i)/q_i] + log (1-p_i)/(1-q_i)}
最后一项与X_i无关,排序时舍去. 得到O(d) = O1 + O2
假设p_i对所有词条是一个小常数,则O1 = \sum_i X_i * log[p_i/(1-p_i)] = c \sum_i X_i
假设文档中绝大多数与查询无关,则q_i = P(w_i) = df_i / N, (1-q_i)/q_i = N / df_i
最终O(d) =\sum_i [X_i * (c + idf_i)]

词条分布模型:
Poisson:假设单词w_i在文档中的出现次数cnt服从Poisson(L_i), L_i = cf_i / D
p(k, L_i) = P(cnt = k) = e^{-L_i} L_i^k / k!
用此分布可进一步估计df_i = N * P(cnt>=1) = N (1 - p(0, L_i))
词义越实际,估计误差越大,因为此时词的出现不再独立
好处:可以用来判断虚词orz..
二重Poisson: 将文档分为两类:此词作为实词(重点,主题)出现的和作为虚词出现的

之前的方法都没有利用词语重现
Latent Semantic Indexing (topic model)
对词条-文档矩阵的SVD

TextTiling: 找到文档中与查询有关的段落
基本思想: 衡量句子的紧凑度(cohesion), 紧凑度与两边的差之和称作深度
紧凑度小,深度大的句子容易成为分割句

较好的方法:Block Comparison. 将前后句子在前后文本块里表示成向量计算距离

#16 Text Categorization
==========
将文本抽象为向量: 将文档单词用词频计算出某种得分
Decision Tree: 按照一列可判定问题选择树的分支
训练决策树: 对于某些独特训练数据,过度训练容易出现overfitting
简单的stoping criterion: 当前节点的所有数据都已具有相同类别
Maximum information gain: 按照某个属性决策,分支信息增益最大
建树后剪枝以优化性能,剪枝会导致训练集上的准确度下降
使用留存数据(held-out data)进行验证/剪枝,是部分不使用的训练数据
为了更充分利用数据,可使用n-fold cross-validation,取一小部分做验证另一部分做训练,循环
决策树清晰易于跟踪.

Maximum Entropy:
固定一个待考察类别,选定K个单词作为base
构造K个特征函数f_i(x,c), x是一篇文档的向量(如取最显著的20维)
若c=1(文档属于目标类别的一维),且x中单词w_i的权重>0,则f_i(x)=1, else =0
loglinear model:
p(x,c) =Z * \prod_(i=1~K){a_i ^{f_i(x, c)} },a_i为待训练参数,表示各个特征的权重,Z normalized
分类时比较p(x,1)与p(x,0)的大小
广义比例迭代法求最大熵模型,要求特征期望相同?
效果很好(96%)
最大熵方法提供了一个整合各个特征的良好框架

Perceptron:
用向量点积与阈值比较做决策,遇到不符合当前模型的样本,就修改此维度上的判断向量和阈值(朝梯度最大方向)
对于linearly separable问题,此学习一定收敛.
但文本分类在word-based的向量空间中不线性可分.(83%)

Nearest neighbor classification:
依赖一个好的相似度计算函数,效率低
特定问题下效果好.

2、《统计自然语言处理基础》的笔记-第121页

区分n-gram数量B与词汇量V

这本书当中依然有很多错误，译者也助长了错误。在第六章 语言模型部分，作者详细定义了各种概念，但是对于B的翻译不够好：训练实例的类别量，其实就是模型的参数数量或者n-gram的数量。围绕这个概念问题，出了一系列错误。

第一个错误表现在127页的译者注释，译者注意到manning公布的勘误表，注意到：“ 训练语料有273266个词形，B应该是273266，。。。译者注 ”。

这里的B应该是V，对于二元语法模型B = V^2。

第二个错误是128页：“ 这个训练语料库共有14585个词形。所以对于新的条件概率p(not|was)，新的估计是(608 + 0.5)/(9404 + 14589*0.5) ”。这里也跟着错，应该是：
(608 + 0.5)/(9404 + 14589^2*0.5)

当然这里是由于原作者错误，译者不察觉。相应地，表格6.5里的ELE估计都是错的，原文结论说折扣掉一半也完全错误。

结论是第六章出现的一系列错误作者难辞其咎。译者也未能指出错误。

http://nlp.stanford.edu/fsnlp/errata.html

page 196, line -13: Change "This will be V^{n-1}" to "This will be V", given the following major clarification: In Section 6.1, the number of 'bins' is used to refer to the number of possible values of the classificatory feature vectors, while (unfortunately) from Section 6.2 on, with this change, the term 'bins' and the letter B is used to refer to the number of values of the target feature. This is V for prediction of the next word, but V^n for predicting the frequency of n-grams. (Thanks to Tibor Kiss <tibor .... linguistics.ruhr-uni-bochum.de>

page 202-203: While the whole corpus had 400,653 word types, the training corpus had only 273,266 word types. This smaller number should have been used as B in the calculation of a Laplace's law estimate of table 6.4 (whereas actually 400,653 was used). The result of this change is that f_{Lap}(0) = 0.000295, and then 99.96% of the probability mass is given to previously unseen bigrams (!). In such a model, note that we have used a (demonstrably wrong) closed vocabulary assumption, so despite this huge mass being given to unseen bigrams, none is being given to potential bigrams using vocabulary items outside the training set vocabulary (OOV = out of vocabulary items). (Thanks to Steve Renals <s.renals .... dcs.shef.ac.uk> and Gary Cottrell <gary .... cs.ucsd.edu>

page 205, line 2-3: Correction: here it is said that there are 14589 word types, but the number given elsewhere in the chapter (and the actual number found on rechecking the data file) is 14585. Clarification: Here we directly smooth the conditional distributions, so there are only |V| = 14585 values for the bigram conditional distribution added into the denominator during smoothing, whereas on pp. 202-203, we were estimating bigram probabilities, and there are |V|^2 different bigrams. (Thanks to Hidetosi Sirai <sirai .... sccs.chukyo-u.ac.jp>, Mark Lewellen <lewellen .... erols.com>, and Gary Cottrell <gary .... cs.ucsd.edu>

《统计自然语言处理基础》

版权说明