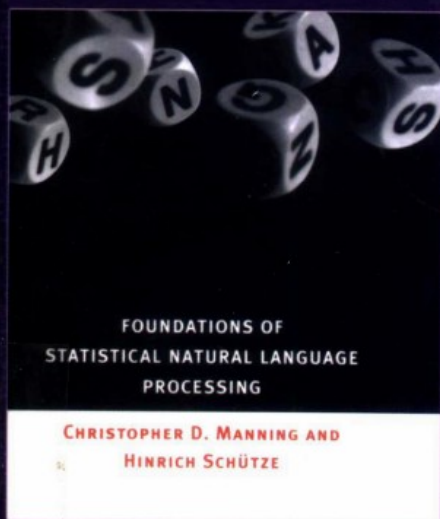


统计自然语言 处理基础

Foundations of
Statistical Natural Language Processing



[美] Christopher D. Manning 著
[德] Hinrich Schütze

苑春法 李庆中 王 昀 等译
李 伟 曹德芳



电子工业出版社
Publishing House of Electronics Industry
<http://www.phei.com.cn>



统计自然语言处理基础

Foundations of
Statistical Natural Language Processing

高采用率的计算语言学权威教材!

近年来,统计自然语言处理(或称统计语言学)异军突起,现已成为自然语言处理研究的主流。

本书是一本全面系统地介绍统计自然语言处理技术的专著,已经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。本书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。为了方便学生 and 研究人员建立自己的语言处理系统,本书不仅提供了宽广而精确的数学和语言学理论基础,还详细讨论了统计语言模型,并覆盖了搭配查找、语义消歧、概率分析、信息检索和其他应用。

相关图书



Christopher D. Manning

美国斯坦福大学计算机科学与语言学助理教授,从事有关人类语言的智能化处理与生成系统及其形式化的研究工作,重点关注语言的概率模型、统计自然语言处理、信息抽取、文本理解和文本挖掘、基于约束的语法(HPSG和LFG)理论及其概率的扩展、句法的类型学、计算词典编撰学(包括XML, XSL和信息可视化方面的工作)以及在计算语言学和机器学习领域里的其他课题。

Hinrich Schütze

德国斯图加特大学自然语言处理学院教授。研究兴趣包括统计自然语言处理、文本挖掘和数据挖掘、信息检索以及文本分类。在斯坦福大学讲授信息检索与文本挖掘课程。

苑春法

清华大学计算机系教授,现主讲研究生学位课程:计算语言学。主要从事计算语言学研究,包括基于语料库的汉语依存语法句法分析、汉语语素库建设、汉语构词研究和汉语金融领域信息抽取等。主持或完成国家自然科学基金、973和863等多项国家项目。

ISBN 7-5053-9921-7



9 787505 399211 >



传递教育信息 共享教育资源
华信教育资源网
www.hxedu.com.cn
欢迎登录 获取优质教学资源



责任编辑:马 岚
特约编辑:马爱文
封面设计:毛惠庚

本书贴有激光防伪标志,凡没有防伪标志者,属盗版图书

ISBN 7-5053-9921-7 定价:55.00 元

国外计算机科学教材系列

统计自然语言处理基础

Foundations of
Statistical Natural Language Processing

[美] Christopher D. Manning

著

[德] Hinrich Schütze

苑春法 李庆中

王 昀 李 伟 曹德芳 等译

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

近年来,自然语言处理中的统计学方法已经逐渐成为主流。本书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。本书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,本书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)。在本书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。

本书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。

Fifth printing, 2002

© 1999 Massachusetts Institute of Technology

Second printing with corrections, 2000

All rights reserved. No part of this book may be reproduced in any form by any electronic or mechanical means (including photocopying, recording, or information storage and retrieval) without permission in writing from the publisher.

Chinese Simplified language edition published by Publishing House of Electronics Industry, Copyright © 2005

本书中文简体版专有出版权由 MIT Press 授予电子工业出版社,未经许可,不得以任何方式复制或抄袭本书的任何部分。

版权贸易合同登记号 图字:01-2002-6457

图书在版编目(CIP)数据

统计自然语言处理基础/(美)曼宁(Manning, C. D.)等著;苑春法等译.

北京:电子工业出版社,2005.1

(国外计算机科学教材系列)

书名原文:Foundations of Statistical Natural Language Processing

ISBN 7-5053-9921-7

I. 统... II. ①曼... ②苑... III. 统计方法-应用-自然语言处理-教材 IV. TP391

中国版本图书馆CIP数据核字(2004)第129789号

责任编辑:马 岚 特约编辑:马爱文

印 刷:北京兴华印刷厂

出版发行:电子工业出版社

北京市海淀区万寿路173信箱 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:27 字数:691千字

印 次:2005年1月第1次印刷

定 价:55.00元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换;若书店售缺,请与本社发行部联系。联系电话:(010)68279077。质量投诉请发邮件至 zltts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。



出版说明

21 世纪初的 5 至 10 年是我国国民经济和社会发展的关键时期,也是信息产业快速发展的关键时期。在我国加入 WTO 后的今天,培养一支适应国际化竞争的一流 IT 人才队伍是我国高等教育的重要任务之一。信息科学和技术方面人才的优劣与多寡,是我国面对国际竞争时成败的关键因素。

当前,正值我国高等教育特别是信息科学领域的教育调整、变革的重大时期,为使我国教育体制与国际化接轨,有条件的高等院校正在为某些信息学科和技术课程使用国外优秀教材和优秀原版教材,以使我国在计算机教学上尽快赶上国际先进水平。

电子工业出版社秉承多年来引进国外优秀图书的经验,翻译出版了“国外计算机科学教材系列”丛书,这套教材覆盖学科范围广、领域宽、层次多,既有本科专业课程教材,也有研究生课程教材,以适应不同院系、不同专业、不同层次的师生对教材的需求,广大师生可自由选择和自由组合使用。这些教材涉及的学科方向包括网络与通信、操作系统、计算机组织与结构、算法与数据结构、数据库与信息处理、编程语言、图形图像与多媒体、软件工程等。同时,我们也适当引进了一些优秀英文原版教材,本着翻译版本和英文原版并重的原则,对重点图书既提供英文原版又提供相应的翻译版本。

在图书选题上,我们大都选择国外著名出版公司出版的高校教材,如 Pearson Education 培生教育出版集团、麦格劳-希尔教育出版集团、麻省理工学院出版社、剑桥大学出版社等。撰写教材的许多作者都是蜚声世界的教授、学者,如道格拉斯·科默(Douglas E. Comer)、威廉·斯托林斯(William Stallings)、哈维·戴特尔(Harvey M. Deitel)、尤利斯·布莱克(Uyless Black)等。

为确保教材的选题质量和翻译质量,我们约请了清华大学、北京大学、北京航空航天大学、复旦大学、上海交通大学、南京大学、浙江大学、哈尔滨工业大学、华中科技大学、西安交通大学、国防科学技术大学、解放军理工大学等著名高校的教授和骨干教师参与了本系列教材的选题、翻译和审校工作。他们中既有讲授同类教材的骨干教师、博士,也有积累了几十年教学经验的老教授和博士生导师。

在该系列教材的选题、翻译和编辑加工过程中,为提高教材质量,我们做了大量细致的工作,包括对所选教材进行全面论证;选择编辑时力求达到专业对口;对排版、印制质量进行严格把关。对于英文教材中出现的错误,我们通过与作者联络和网上下载勘误表等方式,逐一进行了修订。

此外,我们还将与国外著名出版公司合作,提供一些教材的教学支持资料,希望能为授课老师提供帮助。今后,我们将继续加强与各高校教师的密切联系,为广大师生引进更多的国外优秀教材和参考书,为我国计算机科学教学体系与国际教学体系的接轨做出努力。

电子工业出版社

教材出版委员会

主任	杨芙清	北京大学教授 中国科学院院士 北京大学信息与工程学部主任 北京大学软件工程研究所所长
委员	王 珊	中国人民大学信息学院院长、教授
	胡道元	清华大学计算机科学与技术系教授 国际信息处理联合会通信系统中国代表
	钟玉琢	清华大学计算机科学与技术系教授 中国计算机学会多媒体专业委员会主任
	谢希仁	中国人民解放军理工大学教授 全军网络技术研究中心主任、博士生导师
	尤晋元	上海交通大学计算机科学与工程系教授 上海分布计算技术中心主任
	施伯乐	上海国际数据库研究中心主任、复旦大学教授 中国计算机学会常务理事、上海市计算机学会理事长
	邹 鹏	国防科学技术大学计算机学院教授、博士生导师 教育部计算机基础课程教学指导委员会副主任委员
	张昆藏	青岛大学信息工程学院教授

译者序

近年来,统计自然语言处理(或称统计语言学)异军突起,现已成为自然语言处理研究中的主流。在统计自然语言处理学科成长的过程中,有四个因素起着推动作用:

1. 由于计算机硬件的发展,使大容量的存储和高速计算已经成为可能;
2. 由于计算机网络的普及,大量电子文本在网络上的涌现,使语料的获取不再困难;
3. 机器学习学科本身的发展日趋成熟,并在许多领域得到了广泛应用,因此它在自然语言处理中的应用已经成为很自然的事情;
4. 由于自然语言本身的复杂性,即使是语言学家也很难用纯粹的人工规则(或规律)来刻画它,这就迫使我们从实际语料中学习语言规律。

统计自然语言处理的研究涉及了传统自然语言处理的各个方面,例如语言分析、机器翻译、信息检索、文本分类等。可以毫不夸张地说,统计学习方法的引入大大促进了这些领域的研究和发展。目前国内几乎所有著名大学的计算机系都在从事这方面的研究(或开设了类似专业)。但是,系统地讲授或阅读这方面的专著并未得到学术界同行们的重视。在一次学术会议上,某校一位教授深有感触地说,“研究生在校学习期间一定要认真读一本专著。”我们对这位教授的发言深有同感。研究生们一定要看最新的参考文献,包括学术会议文章和杂志文章;但只看这些资料,不看(或学习)一两本专著,所学知识可能是支离破碎的,也未免有急功近利之嫌,尤其是对一些新兴学科更是如此。在这样的情况下的研究往往底气不足,不容易出一些像样的成果。在学术交流中,往往大家没有共同的语言,甚至闹出笑话。

本书是一本系统介绍统计自然语言处理(或统计语言学)的专著,在国外已经被许多大学用来作为教材。在国内,大家已经开始认识到了这本书的价值,不少大学将本书的英文版作为研究生教材。将这本专著翻译并介绍给国内广大从事自然语言处理研究的读者,具有重要的现实意义。本书覆盖了统计自然语言处理的各个领域中最重要主题,内容详尽,层次清楚。无论是对于从事信息检索、机器翻译、文本分类和语言分析等方面的研究的人员,还是对于计算语言学专业的本科生和研究生,本书都有着非常重要的参考价值。

本书由清华大学计算机系的苑春法组织翻译。苑春法长期从事统计自然语言处理相关领域的研究和教学工作,对该领域里的问题有一定深度的了解。参译者也都在该领域里具有一定的研究基础和经历。本书第2章和第13章~第16章由李庆中初译,第1章和第5章~第8章由王昀初译,第3章和第9章~第12章由李伟初译,前言部分和第4章由曹德芳初译。最后,全书由苑春法负责统一修改、审阅并定稿。在翻译本书的过程中,大家力求忠实于原著,在此基础上尽量把概念表达准确、清晰。黄昌宁教授对于本书的翻译工作给予了指导,闻扬、周剑辉、徐薇、翁耀、钱冬蕾和林静等人也做了部分内容的翻译和辅助性工作,在此一并表示感谢。

本书采用英文版第5次印刷的版本进行翻译,已经对照作者在网站上提供的勘误表对相关内容进行了更正或注解。由于译者水平有限,翻译中难免会出现一些不妥之处,希望广大读者批评指正。

前 言

现在是一个在线信息、电子通信和互联网流行的年代,一本详细介绍统计自然语言处理的教材的需求程度可能并非那么迫切。但是,我们应该看到,商业部门、政府机构以及个人正面对着越来越多与工作、生活密切相关的文本信息,而如何从这些大量文本中挖掘潜在的有使用价值的信息,仍然是一个难题。

与此同时,由于大规模的文本语料的可获得性,人们已经改变了语言学和认知科学中研究自然语言的方法理论。以前一些无足轻重的研究领域,以及一些支离破碎,显得很无趣或者难以觉察的语言现象,都逐渐成为当前研究的热点。然而,在 20 世纪 90 年代早期,定量方法还没有引起语言学研究者的足够关注,在当时的一本数理语言学权威教材中甚至根本没有提及这种方法,但现在它已经被看做语言学理论研究中极为重要的手段之一。

本书希望在理论和实践之间以及直觉和严密之间尽可能地达到一种平衡。具体地讲,就是以数学和语言学作为基础来阐述各种理论方法,同时为了避免材料过于枯燥,努力做到理论方法和实际问题的紧密结合。为了给读者奠定必要的基础知识,我们首先介绍概率论、统计学、信息论和语言学方面的重要概念,使读者能够正确理解并增加这些领域的知识;然后介绍统计自然语言处理中存在的问题,比如标注和消歧问题,还将选择一些重要的研究问题进行讲述,从而使读者能够进一步理解语言学研究中的特殊问题,为更加深入的研究工作提供必要基础。

当初我们设计本书的基本结构时,对于应该包含什么素材以及如何组织这些素材都进行了细致的考虑。其中一个重要的标准就是尽量不要使本书篇幅过长(我们没有完全成功做到这一点)。另外,本书并不打算全面地介绍概率论、信息论、统计学和统计自然语言处理中涉及的其他领域的数学知识。但是,我们尽力做到使本书覆盖统计自然语言处理各个领域中最重要主题。对于那些对数学基础有特殊兴趣的读者来说,需要参考本书之外的其他资料来进行更加深入的研究。

我们也尽量避免使用均匀的笔墨来描述统计自然语言处理以及用到的数学工具和理论。虽然一个内在一致的数学理论很重要,但实际上这种理论在这个领域中并不存在,这就导致了一些地方使用了折中的混合理论。但是,我们可以肯定地说,在自然语言处理中的某种方法也许是对的,但就此断定该方法优于其他方法还为时过早。

本书没有包含语音识别的内容也许会让读者有些出乎预料。这样安排主要是因为我们考虑到,对于自然语言处理来说,语音识别是作为一个相对独立的领域从电气工程专业分离出来的,拥有自己的会议和期刊以及自己的相关研究。然而,最近几年随着研究领域的交叉和互相渗透,统计方法在语音识别中的成功应用激发了自然语言处理中应用统计方法的热潮,本书介绍的许多技术方法都是首先在语音识别中应用,然后慢慢扩展到自然语言处理领域的。特别是语音识别中语言模型的有关工作和本书中语言模型的讨论,在很多方面有共同之处。甚至可以这么说,语音识别是自然语言处理领域中当前最为成功而且应用最为广泛的。但是,有些合理的理由把语音识别排除在本书内容之外:已经有一些比较好的关于语音的教材,而且语音

也不是我们专门研究或者特别擅长的领域,况且即使本书不包括语音的内容也显得有些篇幅过长。另外,虽然两者内容有所交叉,但是差别也很明显:语音识别的教材需要包含信号分析和声学模型方面的内容,对于一个具有计算机科学或者自然语言处理背景的人来说,这些内容并非他们感兴趣或者可以理解的;反之,许多研究语音识别的人可能对我们提到的自然语言处理的主题并不感兴趣。

和统计自然语言处理稍微有些关联的其他领域包括机器学习、文本分类、信息检索和认知科学。在所有这些领域中,都可以找到一些本书中没有包含但是却非常适合本书的例子。由于篇幅所限,我们没有包含一些重要的概念、方法和程序,比如最小描述长度、回溯算法、Rocchio 算法,以及和语言处理的频率效应相关的心理学和认知科学文献。

如何严格区分统计和非统计自然语言处理是一件很困难的事情。开始写这本书的时候,我们相信,两者之间有一条很明显的分界线,但是最近这条线变得越来越模糊了。越来越多的非统计学研究者们采用了语料库证据和一体化的定量方法。在统计自然语言处理中,大家逐渐接受了这样一种观点:当处理某种语言现象时,可以使用和该现象相关的所有可获得的科学知识来构造一个概率模型或者其他模型,而不是简单地采用所谓忽视这类已知知识的方法。

许多自然语言处理的研究者们都对单独编写一本统计方法书籍是否必要提出了质疑。在本书中,最后一件工作就是要改变认为语言学理论与符号计算工作和统计自然语言处理无关的错误看法。然而,我们相信,由于需要涉及这么复杂的基础材料,所以很难写出一本篇幅可以控制,让读者满意并且详细介绍所有自然语言处理知识的教材。此外,还存在许多其他很好的文章,如果对统计和非统计的方法之间需要更多的平衡,我们推荐阅读这些补充资料。

最后要说一下本书的书名“Foundations of Statistical Natural Language Processing”。那些从标准统计学了解统计方法的定义的人可能会对书名有些疑问。我们定义的统计自然语言处理由所有的自动语言处理的定量方法组成,包括概率模型、信息论和线性代数。概率论是统计推理的基础,本书中把术语“统计”的基本含义稍微扩大了一点,即包含处理数据的所有定量方法(一个可以在几乎任何词典中确认的定义)。统计自然语言处理在过去 20 年中是使用得最广泛的一个术语,用它来代表自然语言处理中非符号化和非逻辑的工作,尽管有潜在的可能引起模棱两可的理解,但我们还是决定继续使用这个术语。

致谢

在撰写本书的这 3 年中,许多同事和朋友都为早期的草稿做过注释或者提出过建议。我们想向他们表达感激之情,特别是要感谢:Einat Amitay, Chris Brew, Thorsten Brants, Andreas Eisele, Michael Ernst, Oren Etzioni, Marc Friedman, Éric Gaussier, Eli Hagen, Marti Hearst, Nitin Indurkha, Michael Inman, Mark Johnson, Rosie Jones, Tom Kalt, Andy Kehler, Julian Kupiec, Michael Littman, Arman Maghbouleh, Amir Najmi, Kris Popat, Fred Popowich, Geoffrey Sampson, Hador Shemtov, Scott Stoness, David Yarowsky 和 Jakub Zavrel。另外,我们要特别感谢 MIT 出版社的 Bob Carpenter, Eugene Charniak, Raymond Mooney 以及一位不知姓名的审稿者,他们对内容和说明都提出了许多改进建议,本书由于他们的建议在整体质量和可用性方面都有了很大的改进。我们希望即使没有特意致谢,当他们注意到书中有些想法来自于他们的建议后,也会感受到我们的感激之情。

我们也同样要感谢:Francine Chen, Kris Halvorsen 和 Xerox PARC,感谢他们对本书第二作者

的支持;感谢 Jane Manning 对第一作者的爱和支持,感谢 Robert Dale 和 Dikran Karagueuzian 对这本书的设计建议,感谢 Amy Brand 作为编辑对我们的经常性的帮助和协助。

反馈

我们尽力使本书做到通俗易懂、内容广泛且正确,但毫无疑问,在许多地方我们还可以做得更好。我们非常欢迎读者发 E-mail 提出反馈意见:

cmanning@acm.org 和 hinrich@hotmail.com

总之,我们希望本书可以让有潜力的学生获益并得到启发。本书收集了统计自然语言处理领域的许多方法,并且用一种容易理解的方式呈现出来。希望本书能够对这个领域的持续快速发展起到一定的作用。

Christopher D. Manning
Hinrich Schütze
1999 年 2 月

阅读指南

本书适合的读者是将要学习一学期“统计自然语言处理”课程的研究生。对于一学期要包含的内容而言,本书显然提供了过多的资料,但是这也给教师提供了很大的空间,他可以挑选自己认为有用的内容来讲授。本书假设读者已经有了一定的编程经验,熟悉形式语言和符号分析方法。同样,我们假设读者了解一些基本的数学概念,比如集合论、对数、向量、矩阵、求和以及积分方法。事实上,我们的要求也只比高中水平稍高一些。读者可能已经选修过符号自然语言处理方法方面的课程,但是阅读本书并不要求更多的背景知识。在概率、统计和语言学方向,我们试图简要概括所有必要的背景知识,因为以我们的经验来看,很多想学习统计自然语言处理方法的人,以前没有这些领域的相关知识(以后这种情况可能会有所变化)。然而,对于一个学生来说,要想打下适当的基础,还必须学习这些领域的一些补充资料,这样将来才有可能成为一个合格的研究者。

阅读本书或者使用本书来教学的最佳方法是什么呢?本书的内容分为四个部分:基础知识(第一部分),词法(第二部分),语法(第三部分)以及应用与技术(第四部分)。

第一部分主要是数学和语言学基础,后续部分都建立在第一部分的基础上。这里介绍的概念和技术在整本书中都会用到。

第二部分包括了统计自然语言处理中以词为中心的工作。这个部分很自然地从小到复杂,渐进地介绍了语言学现象,分成4章分别讲述搭配、 n 元语法模型、语义消歧和词汇获取,每章也可以独立阅读。

第三部分的4章分别讲述马尔可夫模型、标注、概率上下文无关文法和概率句法分析,它们是依次建立在前一章的基础上的,因此最好按照顺序阅读。但是,关于标注的这一章可以单独阅读,偶尔需要参考一下马尔可夫模型的内容。

第四部分的主题是4种应用与技术:统计对齐和机器翻译、聚类分析、信息检索和文本分类。它们之间很少有关联,所以根据兴趣和时间,这些章节可以分别单独阅读。

尽管在本书的第一部分介绍了很多背景知识和基础资料,但是如果把本书作为教材,我们并不主张在课程开始的时候仔细学习所有章节。作者通常会把一门课程的前6个学时里需要回顾的主要内容写在第一部分。这些必要的内容包括概率(2.1.8节)和信息论(2.2.7节),以及一些必要的实用知识(第4章包含一部分这样的知识)。我们把第3章作为一个阅读作业留给那些没有多少语言学背景的读者。语言学概念方面的知识在许多章节中都需要,特别是第12章,教师讲到这里时最好复习一下句法概念。在授课过程中,把前面其他章节中的一些资料仅仅当做需要了解的内容进行介绍即可。

关于第二部分主题的组织结构,我们尽可能使之满足以下要求:在课程的早期给出一些比较容易理解的主题,特别是可以为学生们写程序、做项目打好基础的主题。搭配(第5章)、语义消歧(第7章)和附着消歧(8.3节)成功地达到了这个要求。我们在本书中很早就介绍附着消歧,表明了语言学概念和结构在统计语言处理中的地位。第6章中有许多详细的参考资料。对于应用(如语音识别或者光学字符识别)感兴趣的读者希望能够包含全部的相关内容,

但是如果 n 元语法模型不是一个特殊的兴趣焦点,读者可能只想读一遍 6.2.3 节。这些对于理解似然性、极大似然估计、两种简单的平滑方法(如果学生想自己构造概率模型,这些方法非常必要)以及评定系统性能的好方法来说足够了。

我们努力提供了丰富的交叉引用,如果需要,教师可以把大多数章节结合前面某处适当的资料独立讲述。特别地,对于搭配、词汇获取、标注和信息检索这些章节,也使用了同样的方法策略。

习题

每一章都附有习题。在难度和范围方面,这些习题相差很多。我们对它们进行了如下的基本分类:

- ★ 简单问题,范围从简单的文本理解到数学变换、简单证明和设想的一些例子。
- ★★ 比较实际的问题,包括编程和语料库研究。其中许多问题都适合作为两周以上的作业。
- ★★★ 大的、难的和开放性的问题。其中许多问题适合作为一个学期的项目。

网址

最后,我们鼓励学生和教师从配套网站上获取更多参考资料。直接访问 MIT 出版社的网址 <http://mitpress.mit.edu>,然后搜索本书即可。

目 录

第一部分 基础知识

第 1 章 绪论	2
1.1 理性主义者和经验主义者的方法	2
1.2 科学内容	4
1.3 语言中的歧义问题是自然语言难以处理的原因	9
1.4 第一手资料	11
1.5 深入阅读	21
1.6 习题	22
第 2 章 数学基础	23
2.1 概率论基础	23
2.2 信息论基础	35
2.3 深入阅读	47
2.4 习题	47
第 3 章 语言学基础	50
3.1 词性和词法	50
3.2 短语结构	57
3.3 语义和语用	68
3.4 其他研究领域	69
3.5 深入阅读	70
3.6 习题	70
第 4 章 基于语料库的工作	72
4.1 基础知识	72
4.2 文本	75
4.3 数据标注	84
4.4 深入阅读	89
4.5 习题	90

第二部分 词 法

第 5 章 搭配	94
5.1 频率	95
5.2 均值和方差	98
5.3 假设检验	101

5.4	互信息	111
5.5	搭配的概念	114
5.6	深入阅读	116
5.7	习题	117
第 6 章	统计推理:稀疏数据集上的 n 元语法模型	120
6.1	Bins:构造等价类	120
6.2	统计估计	122
6.3	组合估计法	136
6.4	结论	140
6.5	深入阅读	141
6.6	习题	141
第 7 章	语义消歧	143
7.1	预备知识	144
7.2	有监督消歧	146
7.3	基于词典的消歧	151
7.4	无监督消歧	158
7.5	什么是语义	160
7.6	深入阅读	162
7.7	习题	163
第 8 章	词汇获取	165
8.1	评价方法	166
8.2	动词子范畴	169
8.3	附着歧义	173
8.4	选择倾向	179
8.5	语义相似性	182
8.6	统计自然语言处理中词汇获取的作用	190
8.7	深入阅读	192
8.8	习题	194

第三部分 语 法

第 9 章	马尔可夫模型	200
9.1	马尔可夫模型	200
9.2	隐马尔可夫模型	202
9.3	隐马尔可夫模型的三个基本问题	205
9.4	隐马尔可夫模型:实现、性质和变形	212
9.5	深入阅读	214
9.6	习题	214

第 10 章	词性标注	216
10.1	标注中的信息源	217
10.2	马尔可夫模型标注器	218
10.3	隐马尔可夫标注器	225
10.4	基于转换的标注学习	228
10.5	其他模型和语言	233
10.6	标注准确率和标注器的应用	234
10.7	深入阅读	237
10.8	习题	238
第 11 章	概率上下文无关文法	241
11.1	概率上下文无关文法的一些特征	244
11.2	概率上下文无关文法的问题	246
11.3	词串概率的计算	248
11.4	内部-外部算法的问题	255
11.5	深入阅读	255
11.6	习题	256
第 12 章	概率句法分析	258
12.1	一些概念	259
12.2	一些方法	280
12.3	深入阅读	287
12.4	习题	289

第四部分 应用与技术

第 13 章	统计对齐和机器翻译	292
13.1	文本对齐	294
13.2	词对齐	303
13.3	统计机器翻译	304
13.4	深入阅读	307
13.5	习题	308
第 14 章	聚类	310
14.1	层级聚类	314
14.2	非层级聚类	321
14.3	深入阅读	328
14.4	习题	329
第 15 章	信息检索	330
15.1	信息检索的背景	330
15.2	向量空间模型	335

15.3	词条分布模型	338
15.4	潜在语义索引	344
15.5	篇章分割	350
15.6	深入阅读	352
15.7	习题	354
第 16 章	文本分类	355
16.1	决策树	357
16.2	最大熵建模	363
16.3	感知器	368
16.4	k 最近邻分类	372
16.5	深入阅读	373
16.6	习题	374
附录 A	统计表	376
参考文献		377

第一部分 基础知识

第 1 章 绪论

第 2 章 数学基础

第 3 章 语言学基础

第 4 章 基于语料库的工作

第1章 绪 论

语言学的目的是为了能够描述和解释我们周围的语言现象,比如对话、写作和其他媒体中的语言现象。人类如何获得、产生和理解语言,如何理解语言表达和物质世界之间的关系,如何理解用于交流的语言结构,所有这些都属于语言学的范畴。为了解释语言的结构,人们设计了一些规则来把语言表达结构化。这种规则方法有很长的历史,至少可以追溯到2000年以前,而在20世纪,语言学家探索更加细化的语法规则,试图描述什么是正确的和不正确的语言表达,他们使规则变得日趋正式和严格。

然而,上述概念存在一个很明显的问题:对于正确的语言表达,我们无法给出一个精确并且完备的特性,因而无法把它们和错误的语言表达清楚地区分开来。实际上 Edward Sapir 已经注意到了这一点,并且提出了著名的格言“All grammars leak”(Sapir 1921:38)。这是因为人们总是扩展和改变规则,以满足他们遇到的语言交流的需要。但是,无论如何这些规则的扩展和改变不是完全没有道理的。例如,一个语言有这样的句法规则:基本英语名词短语是由一个任意的限定词、几个形容词和一个名词组成的。但为了语言使用的灵活性,我们需要放宽某些限制。

本书探求一种方法来解决上面的问题。我们不把句子分类为合乎语法的句子与不合乎语法的句子,而是着手来解决这样的问题:语言使用中通常出现的形式是什么。用来识别这些模式的主要工具是计数,就是通常所说的统计方法,因此概率论是本书的主要理论基础。而且,我们不仅仅把这类问题看做一个学术问题,而是更希望表明怎样建立一个统计语言模型并有效地使用它来处理许多自然语言处理任务。虽然实际应用的方法和基本理论有些出入,但是统计语言模型的有效性在逐渐证明这种基本方法的正确性。

进行统计自然语言处理,需要掌握一定数量的理论工具。在深入钻研大量理论之前,本章将花费一些时间解决统计方法在自然语言处理中的定位问题。因为本书中大量的篇幅都和这种方法有关,所以我们将首先考察一些哲学理论和重要思想,这些都是统计方法用在自然语言处理中的动因。并且,我们首先考察使用统计方法在文本中能学到些什么,获得我们的第一手资料。

1.1 理性主义者和经验主义者的方法

一些语言研究者和很多自然语言处理研究者的工作重点仅仅在文本层面,而不去思考语言表达的意思和语言书写形式之间的关系。对统计方法感兴趣的读者可能更注重实践部分,但是即使是注重实践的工作者,也不得不面对这样的问题:什么样的先验知识应该加入他们设计的语言模型。如果这样的先验知识和他们设想的有很大出入时,是否也要把它们加入到模型中。本节简要地讨论了这个问题的基础,即相关的哲学问题。

大约在1960年至1985年,语言学、心理学、人工智能和自然语言处理中的大部分研究完全被一种理性主义方法(rationalist approach)所支配。这种理性主义方法是由一种信仰决定的,人们相信在人类头脑中重要的知识不是由感官得到的,而是提前固定在头脑中,由遗传基因决定的。在语言学中,理性主义者已经占据了绝对的支配地位,因为人们已经广泛地认同了 Chomsky(乔姆斯基)提出的关于语言本能的观点。在人工智能领域中,理性主义者试图建立一个智能系统,他们

希望在这个智能系统中通过手工编码大量的先验知识和推理机制,得以复制人类大脑中的语言能力。

Chomsky 坚持这种先天结构的论点,因为他观察到了一个刺激缺乏(poverty of the stimulus)的问题,例如(Chomsky 1986:7)。他认为孩子们在幼年的时候听到的语言是一个有限的输入,要观察他们怎样从这个有限的输入中学到像自然语言这样复杂的东西是非常困难的。理性主义假定语言的关键部分是天生的,是生来就存在于大脑中的人类基因遗产的一部分,通过这个假设来回避这个困难的问题。

经验主义方法同样假设大脑中存在某些认知的能力,该方法和理性主义方法的区别不是绝对的,而只是在某种程度上有所区别。理性主义方法假设大脑中存在某些原始的结构,经验主义方法更倾向于相信人类大脑中有一种结构,这种结构能从感官输入的信息中组织和产生语言,因此不经过学习生成某些语言是可能的。但是经验主义方法的突破在于,它认为人类的智能不是开始于细化的规则集,也不是开始于针对各种各样语言结构和其他感知领域的程序集。经验主义者认为婴儿的大脑最初具有联想、模式识别和概括的一般能力,这些能力通过可以得到的丰富的感官输入,使孩子学习到自然语言的详细结构。在 1920 年至 1960 年,经验主义者统治了上面提及的几个领域,并且现在正在复苏。相对于自然语言处理,经验主义方法认为可以通过一个适当的语言模型学习复杂的和广泛的语言结构,通过把统计学、模式识别和机器学习的方法应用到大规模的语言使用例子中,可以得到模型参数的数值。

通常,在统计自然语言处理中实际上不可能研究观测到大规模的语言实例。所以,人们简单地使用文本作为替代,并把文本中的上下文关系作为现实世界中语言的上下文关系的替代品。我们把一个文本集合称为语料库(corpus),corpus 是一个拉丁文简称,当有几个这样的文本集合的时候,我们称之为语料库集合(corpora)。在这种基于语料库的方法中,可以找到英国语言学家 J. R. Firth 提出的经验主义思想的早期主张,他提出了一句口号“你可以由一个词的上下文理解这个词”(You shall know a word by the company it keeps)。然而基于语料库的经验主义方法似乎在美国结构语言学家的工作中更加清晰,特别是在 Zellig Harris 的工作中。例如,(Harris 1951)试图找到一个可以自动发现语言结构的程序。虽然该工作没有通过计算机来实现,甚至似乎有些幼稚,但是我们还是可以看到一些思路,即好的语法描述能为文本语料库提供一种简洁的表述。

在这里给出从有细微不同的哲学观点来表述语言的科学研究方法是不合适的,让我们来注意理性主义和经验主义方法之间的一些更大的差别。理性主义者和经验主义者试图表述不同的事务。乔姆斯基语言学派(或产生式)探寻描述人类头脑中的语言的模型(I-语言),这个语言模型的数据,即文本(E-语言),只提供了间接的证据,这个证据是可以被母语说话者下意识补充的。经验主义方法感兴趣的是描述实际出现的 E-语言。(Chomsky 1965:3~4)因此提出了在语言能力(linguistic competence)和语言性能(linguistic performance)之间的关键区别。语言能力反映了母语说话者脑海中假设存在的语言结构知识,语言性能则受到一系列事物的影响,例如记忆的局限性和环境的传递噪声。产生式语言学同意这样的观点,即认为人们可以孤立地看待语言能力并且单独表示它,但是经验主义方法通常反对这样的观点,而是想刻画语言的实际使用。

这些区别在很大程度上导致了重视计算工作的经验主义方法的复苏。在人工智能的第二阶段工作中(大约在 1970 年至 1989 年),人们关注于智能科学,更准确地说就是希望建立一个小型的智能化系统。由于这些工作,提出了很多人工智能的关键问题,得到了一些现在仍在使用的方法,但是在很多方面还不尽如人意。例如,以这种工作方式仅能处理非常小的问题(这

样的问题通常被轻蔑地称为“玩具问题”),并且不能给出有效的客观评价。现在,人们已经更加专注于实际工程问题的解决,集中精力探寻一些可以作用于现实世界中未经处理的语料的方法,并且给出不同方法的优劣的客观评价。这个新的研究重点有时也可以表现在命名上,人们用“语言技术”和“语言工程”来代替自然语言处理。同时,这些新的研究重点已经趋向于支持统计自然语言处理方法,因为统计自然语言处理在自动学习和消歧方面更加适用,并且它已经在语言学中占有了一席之地。

最后谈谈乔姆斯基语言学派。虽然这种语言学承认在规则之间存在竞争,但是它仍然依赖于绝对规则:一个句子或者合乎语法,或者非法。一般来说,这种语言学派和美国的结构主义语言学派(American structuralism)是相同的。我们在统计语言处理中的研究方法源自 Shannon 的工作,研究的目的是为语言事件指定概率,使我们可以说那些句子是经常使用的和不经常使用的。这样做的结果导致,当乔姆斯基语言学派趋向于研究非常罕见的句子类型的绝对判定时,统计自然语言处理的实践者感兴趣的是发生在总体语言应用中的关联和倾向性。更进一步讲,经验主义者经常发现,通过研究句子经常使用的形式,人们可以很好地刻画真实世界。

1.2 科学内容

本书给出的许多应用方法都具备很强的应用性和可操作性。换句话说,人们对在自然语言处理中使用统计方法抱有的热情,在很大程度上是因为他们看到了统计方法在解决实际问题的前景,而这些问题正是传统方法无法解决的。但是,如果统计方法仅仅是一种实践的工程方法,或者仅仅是能解决(而科学尚无法解决)语言中的一些困难问题的一种近似方法,那么我们对它的兴趣也会大大降低。起初我们就必须强调,大家对语言使用形式的频率感兴趣是有清晰的、毋庸置疑的科学原因的。换句话说,统计学是研究语言的一种方法。

1.2.1 语言学需要回答的一些问题

什么样的问题是语言研究涉及到的问题呢?在开始的时候,我们将回答两个基本问题:

- 人们说的话是什么?
- 人们说/问/要求的事物相应于现实世界的是什么?

从这两个基本问题,我们的注意力很快就扩展到了这样的问题:人类是怎样获取语言知识的?人们事实上是如何实时地生成和理解语言的句子的?现在让我们仅仅关注上面的两个基本问题。第一个问题涉及了语言结构的所有方面,第二个问题涉及了语义学、语用学和话语,即怎样把人类说的话和现实世界连接起来。第一个问题是语料库语言学的研究范畴,而一个词的使用模型可以作为一个替代品,使人类深入理解它的语义和语用,因此我们也可以用基于语料库的技术来处理第二个问题。但是,语料库中含有的语言模型更容易揭示语言的句法结构,所以统计自然语言处理的主要工作就是解决第一个问题:人们说的话是什么?

传统的语言学(结构主义的/产生式的)是怎样尝试回答这个问题的呢?它不是描述人们常说的话语形式,而是寻求描述一个可以成为语言基础的“能力语法”(competence grammar),假设某些产生式的方法在说话者的大脑中。这种方法对于人们说的话是什么的问题仅仅给出了一个合乎语法的句子集合,这些句子合乎能力语法,而除此之外其他的词串是不合乎语法的。

这种语法合理性的概念意味着我们只是单纯地判断一个句子是否是结构完好的,而不管句子是否是人们习惯的表达方式,或者句子是否有正确的语义。Chomsky 给出了一个例句:Colorless green ideas sleep furiously,尽管它的意思古怪并且不符合人们说话的习惯,但它是一个合乎语法的句子。句法结构合理性因此成了一个绝对的二值选择^①。

把句子区分为合乎语法与不合乎语法的句子,初看起来似乎具有其合理性。当一个非母语说话者说错话的时候(不合乎语法),我们可以马上察觉并且能够迅速纠正这些错误。另一方面,除了发音错误以外,母语说话者通常不会产生语法错误。但是,这里至少有两个理由说明我们为什么要继续研究这个问题。第一,虽然在一些简单情况下,合乎语法与不合乎语法的二值划分似乎是合理的,但是当我们把研究规模扩大时,这样的划分就会变得很牵强。第二,即使不考虑这些,我们也有很多理由把概率应用到研究中,因为仅仅简单地把句子划分为合乎语法的和不合乎语法的,无法像概率那样提供更多的信息。例如,非母语说话者说的或写的东西通常都不是完全不符合语法的,这些句子只是稍微有点奇怪。例如一篇学生作文中有:

In addition to this, she insisted that women were regarded as a different
existence from men unfairly. (1.1)

对于这句话,我们可以理解它的意思,但稍微变换一下表述方法听起来会更自然。这就是表达形式的习惯性问题了。习惯就是人们经常说或者经常使用的方法。

句子不可以简单地划分为两个集合:合乎语法的与不合乎语法的句子集。这是具备实际语言学工作经验的人所共知的事实。而且,对于很多在理论语言学中有重要意义的复杂句子,人们很难确定它们是否是合乎语法的。例如,你可以试着判断下面句子的语法合理性,这些句子来源于一本教材(van Riemsdijk and Williams 1986)。

- a. John I believe Sally said Bill believed Sue saw.
- b. What did Sally whisper that she had secretly read?
- c. John wants very much for himself to win.
- d. (Those are) the books you should read before it becomes difficult to talk about.
- e. (Those are) the books you should read before talking about becomes difficult. (1.2)
- f. Who did Jo think said John saw him?
- g. That a serious discussion could arise here of this topic was quite unexpected.
- h. The boys read Mary's stories about each other.

答案:a.OK, b.bad, c.OK, d.OK, e.bad, f.OK, g.OK, h. bad

我们发现,关于这些句子是否合乎语法的看法,大部分人与 van Riemsdijk 和 Williams 的观点有出入。这样的结果提出了一个现实的问题,产生式语言到底描述什么。

^① 在 Chomsky(1980)提出的管辖与约束(Government-Binding,简称 GB)理论的一些版本中给出了一个比较细的打分等级,这个理论认为句子不遵守某些语法约束是可以容忍的,相应的句子只是有点怪异,而如果句子不遵守另一些语法约束则将被认为是不可容忍的。但是在正式的 GB 理论版本中就不再支持这种概念了。语言学家通常根据一个由星号和问号组成的非正式的系统对句子进行分级划分,在划分的结果中*(ungrammatical)>? * >?? >?(questionable)。但是在发展语法规则的时候,这些分级结果又会被转化为一个二值判定(合乎语法/不合乎语法)。

在语言学著作中已经有很多论述提及了语法合理性判断的困难,或者说是判断结果的模糊性。对于每个句子,它们是否合乎语法似乎都应该有一个绝对的答案,但是人们得出这个答案非常困难。大多数的理论语言学家不顾这些明显的困难,思维仍然拘泥于以前的框架内,他们把判定困难的观测结果归为研究领域之外的内容(归为性能影响),而我们认为这是不合理的。另外,应该注意到大多数的简单句是二值可分的(合乎语法/不合乎语法),我们希望我们的理论也应将这些语言现象考虑在内。或许正确的方法是应同时注意“直观感觉”(categorical perception)的情况,直观感觉已经在心理学著作中被描述清楚了。例如,虽然区别“p”和“b”的发音时间的控制是一个连续变量(它的典型值在语言之间是不同的),人类还是能够凭感觉得出惟一的结果,除非在语音产生的过程中所有的变量都发生在连续的空间中。这就是基于绝对音素的语音学理论可行的原因。和句法结构类似,绝对的句法判断理论也可以满足某些用途。但是,对于复杂并且嵌套的句子,给出它的语法合理性判别非常困难,所以我们很难把这样的绝对判别理论推广到语言应用的所有领域中。

1.2.2 语言中的非绝对现象

上面讨论的困难提醒我们,如果考察一下语言的细节问题,就会发现绝对性假设错误的证据,并且还会发现使用概率来理解语言的必要性。这表明,虽然语言的绝对观点可以满足很多用途,但是我们必须看到它有自身的局限性(就像牛顿物理学一样,适用于很多情况,但是有自身的局限性)^①。

语言中的非绝对现象和语言的演变历史有关(也和社会语言学演变和语言发展的竞争性假设有关)。随着时间的流逝,语言中的词汇和语法也相应发生了改变。词汇改变了自身的意思和词性。例如,英语的 *while* 过去通常作为一个专用的名词,意思是 *time*,这种用法现在主要存在于少数的固定词组中,如 *to take a while*,它现在主要被用做引起从句的补语化成分(*While you were out, ...*)。很难绝对地说清楚,在 1742 年的某一天之前 *while* 是一个名词,然后它就变成了一个补语化成分,即使说这个决定仅仅是被某些说话者做出的,而不是由某个语言团体做出的。更确切一些,人们认为这是一个逐渐的改变过程。一种假设认为,如果一个词在不同上下文中的使用频率逐渐改变,那么可以使它离开以前所属范畴的使用方式,并转向其他范畴中更加类似的词的使用方式,然后这个词就会被重新分析为一个不同范畴的词。在这个改变的过程中,人们可以看到非绝对现象的证据。

词性混合: *near*

初看起来, *near* 可以作为形容词,见例句(1.3a),也可以作为介词,见例句(1.3b):

- a. We will review that decision in the near future.
 - b. He lives near the station.
- (1.3)

near 作为一个形容词的证据有两个,首先是它在句子中所处的位置,位于限定代词和名词中间,这是一个形容词的典型位置;其次是它加上 *-ly* 可以形成一个副词: *We nearly lost*。 *near* 作为一个

① 不熟悉语言学和自然语言处理的读者,理解本节内容可能会有些难度,可以跳过本节继续阅读,读完第 3 章之后再阅读本节。本节中所举的语言演化的例子包括古英语拼写到标准英语拼写的演变,了解这些例子有助于解释现在的一些现象。

介词的证据包括它可以在地点词组之前作为动词的补语,正如例句(1.3b)中所示的那样,这是一个介词的典型角色,并且这个词组可以被 *right* 修饰, *right* 通常用来修饰介词短语: *He lives right near the station* (比较 *He swam right across the lake* 和 *That's a right red car*)。虽然迄今为止这些例子并没有让人那么惊奇,因为在英语中很多单词似乎都具有多重词性,例如很多单词都有名词和动词的双重词性,对于 *play*,可以有 *They saw a play* 和 *They play lacrosse on Thursdays*。但有趣的事情是, *near* 可以同时具有形容词和介词的特性,从而具有了混合词性。这种情况可以在下面的例句中看到:

- a. He has never been nearer the center of the financial establishment.
b. We live nearer the water than you thought. (1.4)

比较级形式是形容词(副词)的特性。其他的词性不能形成比较级和最高级^①。另一方面,语法合理性理论告诉我们形容词和名词后不能接直接宾语,因此我们不得不在形容词的后面插入介词,比如 *unsure of his beliefs* 或者 *convenient for people who work long hours*。在这个意义上讲, *nearer* 扮演了一个介词,在地点词组之前并且接一个直接宾语。这样,在这个句子中 *nearer* 就扮演了形容词和介词的特性,因此表现出了这两种词性的混合特征,而它们通常是有绝对的划分的。

语言的变化: *kind of* 和 *sort of*

词串 *kind of* 和 *sort of* 的新用法提供了一个令人信服的例子,说明在某些句型中不同的使用频率怎样导致一些绝对的变化。在现代英语中, *kind of* 和 *sort of* 至少有两种截然不同的用法。在第一种用法里, *sort* 和 *kind* 作为名词, *of* 作为其后面的介词引导一个介词短语,例如 *What sort of animal made these tracks*。但是在其他的用法中,它们被看做是程度修饰词,类似于 *somewhat* 或者 *slightly*:

- a. We are kind of hungry.
b. He sort of understood what was going on. (1.5)

我们可以说,在这里 *kind/sort of* 不是作为一个普通的名词介词序列,因为它出现的上下文(在主语名词短语和动词之间)通常不能插入一个名词介词序列(例如,不能说 * *He variety of understood what was going on*)。

在历史上, *kind* 和 *sort* 是明确无疑的名词。在句子中,它们可以在前面接一个限定词,后面接一个介词短语:

- a. A nette sent in to the see, and of alle kind of fishis gedrynge. [1382]
b. I knowe that sorte of men ryght well. [1560] (1.6)

直到 19 世纪才出现没有歧义的程度修饰用法:

- a. I kind of love you, Sal—I vow. [1804]
b. It sort o' stirs one up to hear about old times. [1833] (1.7)

① 有些读者可能会想到,一些介词也可以形成以 *-er* 结尾的形式,它们似乎就是这些介词的比较级(例如, *upper*, *downer*, *inner*, *outer*),但是我们注意到,这些介词没有和正规形容词类似的最高级,比如 *near* (*nearest*),并且它们的 *-er* 形式也不能作为形容词使用了。我们不能说: * *John lives inner Sydney than Fred*。

这种新的解释看来不是从其他语言中借用过来的,而是在一门语言的内部发展中出现的。这样的变革是怎么产生的呢?

一种似是而非的假设这样认为,当我们在一个形容词前面加上 *kind/sort of* 的时候,事实上就会出现两种歧义的读法:

- a. [NP a [kind] [PP of [NP dense rock]]] (1.8)
 b. [NP a [AP [MOD kind of] dense] rock]

并且人们发现在 16 世纪至 19 世纪,有一个关于 *kind/sort of* 在结构 [Det {*sort/kind*} of AdjP N] 中使用的重要起源:

- a. Their finest and best, is a kind of course red cloth. [c. 1600]
 b. But in such questions as the present, a hundred contradictory views may preserve a kind of imperfect analogy. [1743] (1.9)

注意, *course* 在这里是 *coarse* 的一个变化了的拼写。在这种情况下, *kind/sort of* 填充一个名词(并且其后带介词)槽(slot),而且它同时也填充一个程度修饰槽(具有不同的句法结构)。当这种用法变得更加常用时, *kind/sort of* 通常就被使用在程度修饰槽中;另一方面,从句法上看它也越来越像一个程度修饰成分。而且,这些特殊名词的语义也被简单地认为具有一种程度修饰的含义。这些经常出现的变化似乎也导致了句法范畴的变化,同时 *kind/sort of* 的使用已经扩展到了其他的上下文,比如修饰动词短语。

一般的观点认为,虽然语言的演变可以是突然的(因为内因或外因),但是通常来说它是逐渐改变的。为了搞清楚渐变的细节问题,我们只有考察使用的频率,感觉词语之间的关系改变的程度,并且这种模型化的过程与绝对化理论相反,需要统计观察。

虽然现在只有少数人的观点试图用统计语言学的理论解释复杂的语言学现象,但是从理论语言学的观点出发,本书感兴趣的是,这是一种考查语言现象的新方法,它可以应用于一些领域,如非绝对现象和语言的演变,并且效果比现在任何一种方法都好。

1.2.3 语言和认知是随机现象

把概率作为理解语言的一种科学方法的最基本论点认为,人类的认知是随机的,因此语言也必须是随机的,因为它是认知的一个完整部分。前面提到的语言中非绝对化现象的例子存在很多疑问,人们认为这些现象都是边缘的和稀少的。在语言中,大部分句子都是二值可分的(合乎语法/不合乎语法),而且在大多数情况下,单词只使用一个词性,没有词性混合的现象。但是如果语言和认知作为一个整体,最好的解释是概率,那么解释语言理论的中心内容就必须是概率论。

用概率的方法去认知世界是因为我们生活的世界中充满了不确定的和不完整的信息。为了能够和世界有效地相互作用,我们需要处理这种类型的信息。假设你想确定渡过一条河是否是安全的,你看到了河水流动缓慢,所以可能它不会把你冲走。而且,你要充分确定河中是否有鳄鱼。你需要综合所有这些信息来判断通过这条河是否安全。现在,如果有人告诉你:“如果你一直向着对岸的大树走,那么河水只有膝盖这么深”,那么这个语言信息就是用来综合考虑的另一个信息源。处理这些单词,了解这句话的全部语义,并且把它作为影响决策的一个因素,这个处理过程大体上和考虑水流并由水流速度形成主意,然后把它作为决策的一个方面没

有什么区别。所以讨论的主旨就是,语言的认知过程和其他的感官输入形式或其他知识的认知过程都是相同的,至少是非常相似的。这些认知过程可以被很好地形式化为随机过程,或者至少限定在一个可以处理不确定和不完整信息构架的量化框架内。

是否赞同定量的方法在语言学中的重要作用,会对语言事实产生完全不同的看法。一个著名的例子就是 Chomsky 的名言,“概率论不适用于语法描述的形式化”。他认为从一个语料库中计算句子的概率将会给每个句子指定一个相同的低概率,合乎语法的与不合乎语法的句子几乎一样,因此这种方法对语言学的发展没有意义,参见(Chomsky 1957:16)。这种观点一般而言仅仅对那些对概率表示有偏见的人来说是有意义的。我们可以考虑一下对“高”这个概念的认知表示。假设你看到一个 7 英尺高的人,他是你见过的最高的人。你将会很轻松地认知出这个人很高,而不会认为无法判断这个人的高矮。类似地,我们也很容易识别出其他未经验证高度的人,比如说一个 4 英尺的人,你肯定不会认为他很高。在本书中,我们将会考虑概率模型,它能够很容易地学习和表示这种类型的规律,并且对未经验证的例子做出正确的判断。更进一步说,统计语言学的主要内容都来源于对未知事件给出一个好的概率估计。前提是不存在这样的假设:在一个概率框架中所有未知事件都被认为是相似的。

我们相信,在人们对概率模型应用于语言(和一般的认知)所持有的怀疑态度中,大部分是因为众所周知的早期概率模型(在 20 世纪 40 年代和 20 世纪 50 年代开发的)非常简单。因为这些简单的模型无法适应人类语言的复杂性,所以它们表现出了概率模型对处理语言的不适用性。我们在本书中推崇的观点之一是,复杂的概率模型可以像一个复杂的非概率模型一样解释问题,并且还带有更多的优点,它们可以解释在认知中和在实际的语言中经常出现的不确定的、不完整的现象。

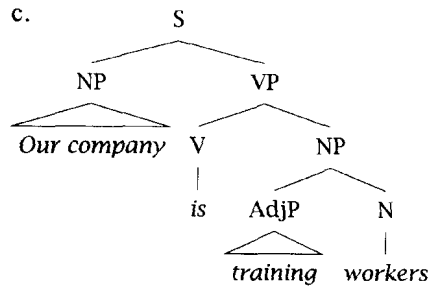
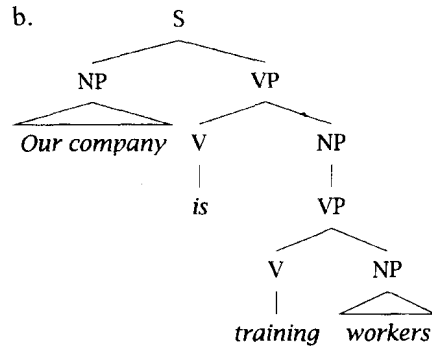
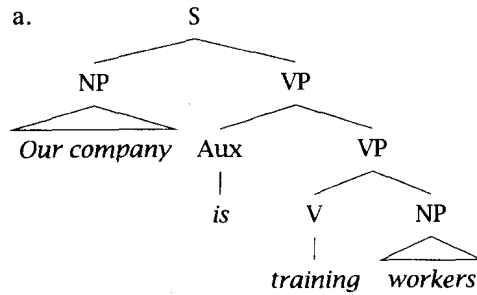
这些问题涉及到了统计自然语言处理中的语义学处理方法。我们先前表述过,在统计自然语言处理中的大部分现有工作都重点关注低层次的语法处理,但是人们有时也会提出这样的疑问,统计方法是否可以处理语言的意思。回答这个问题的主要困难是我们把这个“意思”定义成什么!如果把意思认为是某些语言中的符号表示,那么统计方法实际上就是有用的,比如当我们把英语翻译成一种 SQL 之类的数据库查询语言时。这种语言当然可以使用统计自然语言处理系统来实现(在第 13 章中将会讨论翻译的过程)。但是,从统计自然语言处理的角度看,会更加自然地认为句子的意思包含在话语上下文的字里行间。从哲学的角度看,我们更加认同 Wittgenstein 在后期著作(Wittgenstein 1968)中的观点,他认为一个单词的意思可以由它的使用环境来决定,参见本章开头的内容。在这个概念前提下,很多统计自然语言处理的研究都直接针对词语意思。

1.3 语言中的歧义问题是自然语言难以处理的原因

一个自然语言处理系统需要确定文本的结构性问题,它至少要能够回答:“谁对谁做了什么?”传统的句法分析系统试图仅仅根据一些可能的结构来回答这个问题,这些结构可以认为某些词在属于某一词性的情况下是合乎语法的。例如,给定一个合理的语法,一个标准的自然语言处理系统认为例句(1.10)具有 3 种句法分析结果,这些结果经常被称为 parse。

Our company is training workers. (1.10)

3 种不同的句法分析结果表示在分析树(1.11)中。



(1.11)

在分析树(1.11a)中,一些人认为 *is training* 是动词词组,在其他两种情况下 *is* 是主要动词。在分析树(1.11b)中 *training* 是动名词(对照 *Our problem is training workers*),但是在分析树(1.11c)中 *training* 修饰 *workers*(对照 *Those are training wheels*)。后面的两种句法分析在语义上是异常的,但是当前在大多数系统中语义分析是在句法分析之后完成的。这就是说,当我们遇到了长一些的句子和复杂一些的语法的时候,歧义问题就会导致句法分析结果数量的成倍增长。例如, Martin et al. (1987)表明所用系统对下面的例句(1.12)^① 给出了 455 种句法分析结果。

List the sales of the products produced in 1973 with the products produced in 1972. (1.12)

因此,一个实用的自然语言处理系统必须具有良好的消除歧义的功能,它要解决词义、词类别、句法结构和语义范畴的歧义问题。但是,结果歧义最小和最大化语法的作用范围这两个目标对于符号自然语言处理系统是矛盾的,因此把语法的作用范围扩大化会导致对于一般句

① 也可以参见 Church and Patil(1982)中类似的例子。

子的不正确的句法分析结果数量增加,反之亦然。更进一步讲,使用人工智能的方法进行句法分析和消除歧义,探寻一种可以深入理解自然语言的模型的经验表明,手工编码句法限制和优选规则是一项非常耗费时间的工作,而且如果语言中大量使用比喻修辞,这样的系统很脆弱(Lakoff 1987)。例如,使用选择约束(selectional restrictions)的传统方法,动词 *swallow* 需要一个动物作为它的主语和一个实体作为它的宾语。但是这样的限制不能接受 *swallow* 的常用且直观的比喻修辞用法扩展,如下所示:

- a. I swallowed his story, hook, line, and sinker.
 - b. The supernova swallowed the planet.
- (1.13)

依赖于手工创建规则和手工调整消除歧义的策略,产生了知识获取的瓶颈问题,并且应用在自然文本的时候评价效果依然很差。

统计自然语言处理(NLP)自动从语料库中学习词汇和结构偏向性信息,以此来探寻解决上述问题。不再单独使用句法的类,例如词性,来进行句法分析。我们认为存在着大量词汇之间的关系信息,也就是说,一些单词趋向于互相组成词组。这种搭配知识可以反映更为深入的语义关系。而且,统计模型的使用可以很好地解决歧义问题:统计模型具有鲁棒性和概括性,并且在含有错误的数据和新数据中性能优异。因此,统计自然语言处理引导产生了一种新的方法,可以在分析自然文本的大规模系统中成功地消除歧义问题。此外,统计自然语言处理模型的参数通常可以在文本语料库中自动估计出来,这种自动学习方法不仅减少了人类在建立自然语言处理系统时的工作量,而且还提出了一个有趣的科学问题,涉及到了人类语言的获得。

1.4 第一手资料

1.4.1 词汇资源

我们已经了解了统计自然语言处理具有的这么多优点,应该怎么进行下一步的实际行动呢?首先,我们需要得到一些词汇资源:机器可读的文本、词典、类义词典和处理它们的工具。这里要简要介绍一些重要的资源,因为在全书中都将涉及到它们。读者可以借助网页得到更多的信息,可以了解如何实际获得这些资源。

Brown 语料库可能是知名度最高的语料库了。它是一个约有 100 万个经过标注的词的语料库,由 Brown 大学在 20 世纪 60 年代和 20 世纪 70 年代整理得到。而且它是一个平衡语料库。也就是说,它试图建立一个可以代表当时美国英语的典型语料库。它覆盖的类型有报道、小说、科技文章、法律文章和很多其他文本。遗憾的是,必须付费才能得到 Brown 语料库,但是它对于以研究为目的的使用收费并不是很高。很多自然语言处理的研究机构都有一个可以使用的副本,所以我们得到这个语料库并不困难。Lancaster-Oslo-Bergen(LOB)语料库是 Brown 语料库的一个英国英语的复制版本。

Susanne 语料库是一个 130 000 个词的 Brown 语料库的子集,它是可以免费使用的。并且,它还标注了句子的句法结构信息,而 Brown 语料库只是在词一级上进行了消歧标注。大规模地标注句法结构的语料库是 Penn 树库。它的文本来源于“Wall Street Journal”(华尔街日报)。这个语料库被广泛使用,但它也不是免费的。

“The Canadian Hansards”(加拿大议会议事录)来源于加拿大国会报告,是最著名的双语语料库。双语语料库是包括两种或更多种语言对照翻译的文本语料库。多语言间的平行(对照翻译)文本对于统计机器翻译和其他的跨语言自然语言处理工作是非常重要的。这个语料库也是必须付费的有偿语料库。

除了文本之外,我们还需要词典。WordNet(词网)是一个英语电子词典。词汇被组织到一个网络层次中。每个节点由相近意思的词集组成。在词汇之间还定义了一些其他的关系,例如分裂或部分-总体关系。词网是免费的并且可以从 Internet 下载得到。

▼ 关于语料库的更多信息详见第 4 章。

1.4.2 词的记数

当我们下载了一些文本之后,底层如何表示,如何分类和处理这些文本都是我们感兴趣的事情。其中的很多内容将在第 4 章中深入讨论。就目前而言,我们假设这些文本都被表示成单词序列。为了本节研究的需要,我们将会使用马克·吐温的著名小说《汤姆历险记》(Tom Sawyer)作为例子。

首先有一些明显的问题需要解决。文本中最常用的词是什么? 答案显示在表 1.1 中。注意,在这个表中占统治地位的少数英语单词具有重要的语法作用,并且它们通常作为一些功能性词汇,例如限定代词、介词和补语化成分。表中惟一例外的词就是 *Tom*, 它的出现频率清楚地反映了我们选择的这本书的特点。这是一个关键点。通常来说我们得到的结果依赖于所用的语料库和样本。人们使用大型的和各种各样的样本集,试图避免像上面提到的这种不正常的结果,但是通常来说,使用真正具有典型性的可以代表英语全部用法的样本是不现实的,而且语料库将会反映构建它的材料。例如,如果语料库包括语言学研究论文,那么像 *ergativity*, *causativize* 和 *lexicalist* 这样的词将会出现,否则它们不太可能在语料库中出现,无论语料库的规模有多大。

表 1.1 “Tom Sawyer”中的常用词

词	使用频率	用法
the	3332	determiner (article)
and	2972	conjunction
a	1775	determiner
to	1725	preposition, verbal infinitive marker
of	1440	preposition
was	1161	auxiliary verb
it	1027	(personal/expletive) pronoun
in	906	preposition
that	877	complementizer, demonstrative
he	877	(personal) pronoun
I	783	(personal) pronoun
his	772	(possessive) pronoun
you	686	(personal) pronoun
Tom	679	proper noun
with	642	preposition

在一篇文本中有多少单词呢?这个问题可以有两种解释。有多少词次(word token)可以解答文本长度的问题。本例中的文本中共有 71 370 个词次(token),所以相比于标准语料库它是一个非常小的语料库,但是已经足够说明一些基本的点。虽然“Tom Sawyer”是一本合适长度的小说(大概比通常的文本少 50 万个字节),但是为了统计语法有宽的覆盖面,我们常常搜集的文本大小要比这个文本的量级大一点。问题的另一个解释是,问有多少个不同的词,换句话说,在文本中出现了多少个种词形(word type)?在上面提到的例子中共有 8018 个不同的词出现。对于这篇文本的大小来说,这确实是一个非常小的数字,这大概是因为“Tom Sawyer”是用通俗文体写的儿童读物(例如,一个相同大小的新闻文本大约包含了超过 11 000 个不同的单词)。一般可以这样来解释 token 和 type。token 是出现的总次数, type 是出现的不同事物的个数。我们可以计算 token 对 type 的比值,它表示每个 type 出现的平均次数,对于“Tom Sawyer”来说,这个比值大约是 8.9^①。

上面的统计告诉我们,在这个语料库中每个词平均要出现 9 次。但是统计自然语言处理中的一个最主要的问题是,词汇的出现次数非常不平均。表 1.2 显示了一些词出现的频率信息。有些词非常常用,出现大于 700 次,并且因此这样的词出现的次数占小说中总词次的 1% 以上(在表 1.1 中有 12 个这样的词)。总之,最常用的 100 个词的出现次数约占文本总单词数量的一半多一点(50.9%)。从另一个极端讲,在语料库中大约有一半的单词(49.8%)只出现了一次。这样的单词被称为“罕用语”(hapax legomena), hapax legomena 是希腊语,表示只用过一次。把这样的单词的范围再扩大一点,会发现绝大多数单词出现的次数非常稀少,超过 90% 的单词出现的次数小于或等于 10 次。然而,出现次数非常少的词占据了相当可观的文本比例,文本中大约 12% 的部分是由出现 3 次或少于 3 次的词组成的。

表 1.2 “Tom Sawyer”中词频的频率信息

词频	词频的频率
1	3993
2	1292
3	664
4	410
5	243
6	199
7	172
8	131
9	82
10	91
11~50	540
51~100	99
> 100	102

像这样简单的文本计算可以在密码学中有一些应用,或者给出文本体裁或作者身份识别

① 单独来看,这个比值并不像“文本复杂度”(text complexity)那样是一种有效的衡量方法,因为它和文本的大小有关。为了得到有效的比较,需要规范化文本的长度,比如在一个 1000 个词大小的窗口中计算这个比值。

的一些提示。但是,这种基于文本词汇分布的粗糙统计很难对语言学有重要的意义,所以我们在本章最后引入了一种研究方法,它的语言学特点更多一些。上面这些粗糙的文本统计计算已经使我们知道了统计自然语言处理的困难:对于在语料库中没有出现或者几乎不会出现的词,我们很难预测出它们的行为。我们最初认为使用一个更大的语料库就可以解决这样的问题,但是这个愿望是无法证实的。下面将要讨论语料库语言学中最著名的早期结论:Zipf 法则,它针对的问题就是这些稀有词汇。

1.4.3 Zipf 法则

在 Zipf 的“Human Behavior and the Principle of Least Effort”一书中,他认为找到了一个统一的原理,即最小精力付出原理,它在本质上是整个人类活动的基础。最小精力付出原理认为人类将会尽可能最小化他们可能的平均工作率(也就是说,不仅仅要最小化我们马上要做的工作,而且要考虑短期内最小工作对未来工作带来的影响)。这种理论的依据是 Zipf 找到的一些经验法则,而且在他自己的研究工作开始的时候就提出了这些法则,目的是揭示语言中的某些概率分布。在这里我们不对 Zipf 的一般性理论进行评述,但是要涉及到一些经验语言法则。

著名法则:Zipf 法则

如果我们能够统计一种语言中所有的词在一个大型语料库中的出现次数,并且按它们出现次数的大小顺序把这些词排列起来,会发现一个词出现的频率 f 和它的排列位置之间的关系,设它的排列位置为 r 。那么 Zipf 法则可以表示成:

$$f \propto \frac{1}{r} \quad (1.14)$$

或者可以这样表示:

$$\text{存在一个常数 } k \text{ 使得 } f \cdot r = k \quad (1.15)$$

例如,可以这样说,排在第 50 位的词的出现次数大约是排在第 150 位的词的出现次数的 3 倍。Estoup 在 1916 年首先发现了这种出现次数和排列位置之间的关系,但是 Zipf 正式定义并推广了这种关系,并且以他的名字命名了这个关系。我们实际上不把这种关系看做一个规则,而是作为某些试验事实的一个比较粗糙的特性。

把 Zipf 法则应用到语料库“Tom Sawyer”上,可以得到表 1.3 列出的评测结果。从表中的数据可以看到,Zipf 法则是近似正确的,但是我们注意到出现次数最高的 3 个词和规则的预期稍有出入,并且更进一步的分析表明,在排名前 100 的词中 f 乘以 r 的结果逐渐增大,这种细微的增长在 Zipf 的很多研究中可以发现。然而,如果把 Zipf 法则看做人类语言中词语概率分布的粗略描述,它还是有用的,人类语言的词汇中有少量非常常用的词,还有一定数量的半常用词和大量的不常用词。Zipf 看到了这个规律的重要性。按照他的理论,说话者和听者都试图去最小化他们的精力付出。说话的人用一个小的常用词表来减少他们的精力付出,而听者则用一个大的冷僻词表来减少他们的精力付出(这样使信息的歧义变小)。这两个矛盾需求中间的最佳折中方法就是词汇的出现次数和排列位置之间的倒数关系。然而,对于我们来说,Zipf 法则的主要结果就是解决了一个实际问题;大部分词汇在我们使用的数据中是极端稀少的,我们仅仅对于少量的词汇有大量的实例。

对于 Zipf 法则的有效性和可能性,Mandelbrot(1954)做了深入的研究和推导。虽然在対大

型语料库的研究中,有时的结果会比我们给出的例子更接近 Zipf 法则的预测,但是 Mandelbrot (1954: 12) 也注意到:“bien que la formule de Zipf donne l'allure générale des courbes, elle en représente très mal les détails(虽然 Zipf 公式给出了这些曲线的一般形式,但它还是很难反映细节情况)”。图 1.1 显示了 Brown 语料库中词汇排列位置和出现次数的曲线(在两个对数轴上)。按照 Zipf 法则的预测,这条曲线应该是斜率为 -1 的直线。Mandelbrot 发现情况并不是这样,特别是在排列位置越低和越高的情况下,曲线偏离预测值就会越严重。在我们的例子中,曲线在大部分低排列位置时偏低于预测值,在大于 10 000 的排列位置时偏高于预测值。

表 1.3 Zipf 法则在“Tom Sawyer”语料库上的实验评价

单词	频率(f)	排名(r)	$f \cdot r$	单词	频率(f)	排名(r)	$f \cdot r$
the	3332	1	3332	turned	51	200	10 200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10 440	Could	2	4000	8000
two	104	100	10 400	Applausive	1	8000	8000

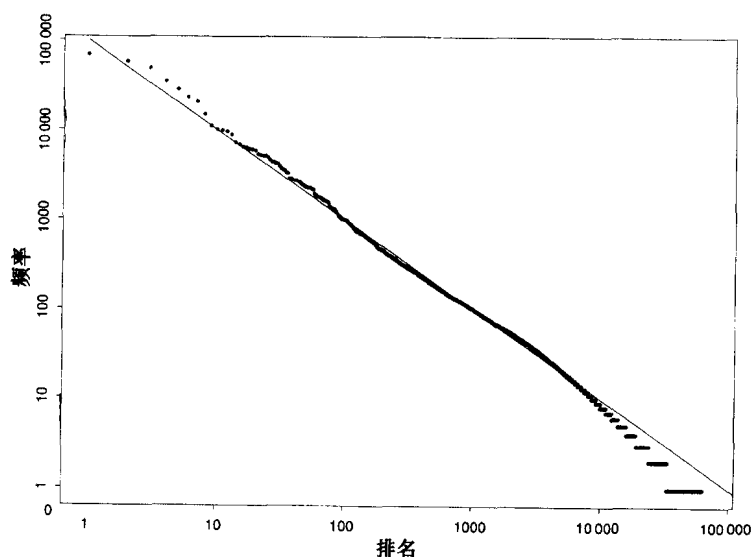


图 1.1 Zipf 法则。图中 X 轴是排列位置, Y 轴是出现频率, 使用对数数值。点线表示在 Brown 语料库中排列位置和出现次数的关系, 直线表示 $k = 100\,000$, 即 $f \times r = 100\,000$ 时的 Zipf 法则曲线

为了得到更加接近词汇经验分布的结果, Mandelbrot 得出了下面的更一般的排列和出现次数的关系:

$$f = P(r + \rho)^{-B} \text{ 或 } \log f = \log P - B \log(r + \rho) \quad (1.16)^{\text{①}}$$

在公式中, P , B 和 ρ 是文本的参数, 它们总体衡量了文本中词汇使用的广度。在排列位置和出现次数之间仍然是一个双曲线分布, 和最初的式(1.14)一样。如果在双对数轴上表示这个公式, 那么对于高排列位置的情况, 这条曲线非常近似于一条以负 B 为斜率的递减直线, 就像 Zipf 法则一样。然而, 通过适当调整其他参数, 可以模拟出一条曲线, 它对大部分常用词的出现频率的预测偏低, 但在曲线的上部仍然有凸起部分, 和我们在例子“Tom Sawyer”中看到的情况一样。从图 1.2 可以看出, 对于我们的语料库, Mandelbrot 的公式比 Zipf 法则更加吻合实际的观测值。左上角微小的凸起和更大的斜率 ($B = 1.15$) 使曲线在最低和最高排列位置时的预测比图 1.1 所示的 Zipf 法则更好。

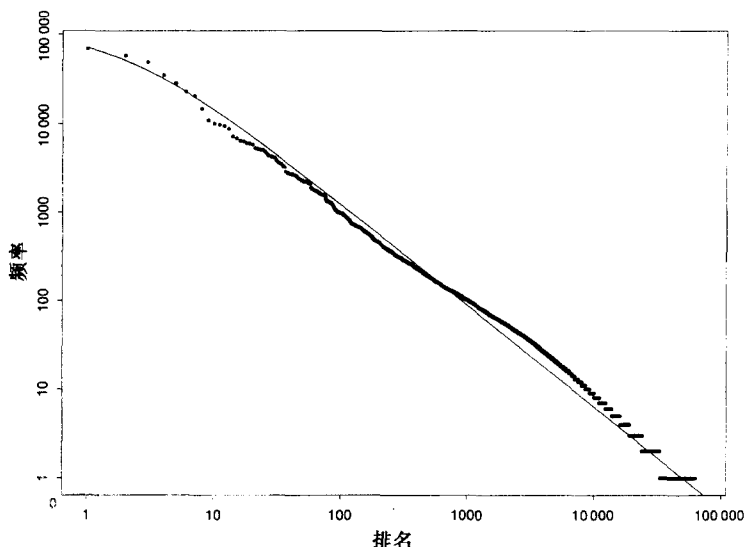


图 1.2 Mandelbrot 公式。图中 X 轴是排列位置, Y 轴是出现频率, 使用对数数值。点线表示在 Brown 语料库中排列位置和出现次数的关系, 直线表示 $P = 10^{5.4}$, $B = 1.15$, $\rho = 100$ 时的 Mandelbrot 公式的曲线

如果我们设置参数 $B = 1$, $\rho = 0$, 那么 Mandelbrot 公式就简化为 Zipf 法则(参见习题 1.3)。根据类似数据, Mandelbrot 得出的结论认为 Zipf 的简化公式在一般情况下是不对的: “Lorsque Zipf essayait de représenter tout par cette loi, il essayait d’habiller tout le monde avec des vêtements d’une seule taille” (当 Zipf 试图利用他的理论描述每件事情的时候, 他试图给每个人都穿上同一款式的衣服)。然而, Mandelbrot 看到了 Zipf 工作的重要性, 他强调了世界上许多现象不符合高斯分布, 而是符合双曲分布。这个事实最早是被经济学领域的 Pareto 发现的。

其他一些法则

统计自然语言处理著作中涉及 Zipf 法则时总是提到上面的法则, 但实际上 Zipf 提出了其他许多关于语言的经验法则, 它们也可以证明最小精力付出原理。其中, 至少有两个法则涉及到了统计自然语言处理关心的问题。其中一个法则提出词汇语义数量和出现频率有关。Zipf

① 本书中的公式均按原著的表示方法, 若无特别指明, 都假设 2 为对数的底数, 同时假定 $0 \log 0 = 0$ 。参见本书第 36 页。——编者注

认为,对话时说话者希望只用一个词就可以表示所有的意思,而听者希望每个意思都用不同的词汇来表达。假设双方的愿望同等强烈,Zipf 认为一个词的语义数目 m 服从下面的法则:

$$m \propto \sqrt{f} \quad (1.17)$$

或者,也可以是下面的形式:

$$m \propto \frac{1}{\sqrt{f}} \quad (1.18)$$

Zipf 提供了支持这个法则的经验数据(在他的研究中,出现次数排在 10 000 左右的词平均有 2.1 个语义项,排在 5000 左右的词平均有 3 个语义项,排在 2000 左右的词平均有 4.6 个语义项)。

第二个法则提出实词有聚集在一起的趋向。我们可以检测一个实词在一篇文本中每次出现的行数和页数,然后计算这个词之间不同的间隔 I 出现的次数 F 。Zipf 发现,对于有 260 000 个词汇的语料库来说,在语料库中至多出现 24 次的词有这样的规律:对于某一个固定大小的间隔,它出现的次数和间隔大小成反比($F \propto I^{-p}$,在 Zipf 的研究中,公式中的参数 p 在 1~1.3 之间变化)。换句话说,大部分情况下同一个实词趋向于在相邻的位置出现。

▼ 关于语义的问题将在第 7 章讨论,关于实词聚集的问题将在 15.3 节讨论。

Zipf 的其他一些法则包括,词频和它们的长度成反比关系,一个词或者语素的出现频率越高,那么它所使用的变换形式也就越多(粗略地讲,变换形式是指复合和词缀变化的复杂形式),而且更进一步的法则还包括了历史演化和音素频率。

强法则的重要性

作为对 Zipf 法则最后的评述,这里有一个争论,这个令人惊奇的、有趣的“强法则”通常是怎样描述自然现象的。我们已经知道随机产生的文本是遵从 Zipf 法则的,参见(Li 1992)。为了证明这一点,我们构造了一个文本生成器,它可以从字母表的 26 个字母和空格之间随机地产生字符(也就是说,这 27 个字符中的每一个都有相同的生成概率)。简单地讲,产生一个长度为 n 的词的概率是 $\left(\frac{26}{27}\right)^n \frac{1}{27}$,即产生 n 个非空格字符并在此之后产生一个空格字符的概率。

人们可以看到,由这样一个生成器产生的词汇服从 Mandelbrot 设计的最小精力付出原理。这是因为,(1) 长度为 $n+1$ 的词的数量应该比长度为 n 的词的数量多 26 倍;(2) 长度为 n 的词要比长度为 $n+1$ 的词更加频繁地出现,并且它们出现概率的比值是一个常量。这两个相反趋向可以被 Mandelbrot 法则结合到一起,参见习题 1.4。

应用计算 Zipf 分布时所用的同一过程,会发现遵从强法则的一个广泛的概率分布类别。首先计算事件的数量,然后按照它们出现的频率进行排列,参见(Günter et al. 1996)。从这个角度来看,Zipf 法则作为语言的一个特性似乎缺少说服力。但是,它的基本观点应该保留:概率方法应用于语言学的困难是几乎所有的词都很稀有。Zipf 法则很好地反映了这种观点。

1.4.4 词的搭配

词典编撰者和语言学家长久以来都对词的搭配很感兴趣。一个搭配是一个短语或者一个可接受的习惯用法,并且它的整体意思不是各部分意思的简单合并。词的搭配包括复合结构(*disk drive*)、动词短语(*make up*)和其他一些固定词组(*bacon and eggs*)。它们经常含有特殊的意思或者是习惯用法。例如,在澳大利亚,书面语言中的官僚主义者(*bureaucrats*)的最佳表达

方式是 *international best practice*。现在看这种表达似乎是不符合常用语法的,它只是两个简单的形容词修饰一个名词而已。但是,当这个短语频繁使用,并且作为某些隐含意思的固定表达方式的时候,它就被认为是一个搭配对。甚至,对于任何一种表达方法,人们由于听到其他人使用它,也开始重复使用它,那么这种表达方式也可以作为一个搭配的候选。

- ▼ 关于词的搭配我们将在第 5 章中详细讨论。我们将会看到搭配在统计自然语言处理领域中的重要性,比如在机器翻译(参见第 13 章)和信息检索(参见第 15 章)方面。在机器翻译中,一个词根据它出现的搭配对不同可以有不同的翻译方式。一个信息检索系统可能只想索引那些“感兴趣的”短语,也就是说,索引的短语最好是搭配对。

词典编撰者也对搭配感兴趣,有两方面的原因,一是因为这些搭配可以显示词汇的通常用法,二是因为搭配是多个词的组合单元,可以独立存在并且可能会进入词典。另外也有理论方面的原因:就大部分语言使用来说,语言就是人们经常听到并重复使用的短语和结构,这样的事实反驳了乔姆斯基学派强调的语言使用由人创造的观点;并且 Hallidayan 的方法对这个观点给予了更多的支持,他认为语言不能从实际使用的环境和社会背景中剥离开。

搭配的组成可以是几个词(例如 *international best practice*),也可以是省略形式的(例如 *make [something] up*),但是我们只是研究最简单的两词情况,并且想知道如何自动识别相邻的两个词是否是一个搭配。前面已经叙述过,搭配的使用比较频繁。所以,为了识别搭配对,我们首先的想法是在文本中简单地寻找最常出现的词序列。这是很容易实现的,我们从纽约时报中抽取文章形成一个语料库,对于这个语料库的实验结果显示在表 1.4 中。遗憾的是,这种方法似乎效果并不好,它不能有效地从文本中找出搭配对。表中出现的这些词对,通常称为二元组(bigram)并不令人惊奇。它们描绘了常用的句法结构,这些结构包括了个别极端常用的词汇。表中给出的 *the*, *of* 和 *in* 都是极端常用的词汇,而且介词和名词短语的句法结构决定了限定词通常出现在介词的后面,所以我们会经常看到 *of the* 和 *in the* 这样的结构。但是这些词串并不是搭配对。没有把组成搭配对的词汇的频率归一化(normalizing)可能就是产生这样的结果的一个原因。在搭配获取的过程中,下一步明显需要做的就是用某种方法把每个词的频率都考虑进去。我们将会在第 5 章中讨论怎样实现这种方法。

对上述方法的一个非常有效的改进是过滤这些搭配,删除这些和我们感兴趣的搭配结构不一致的候选。我们不感兴趣的搭配结构有很多,比如一个介词后面接一个定冠词的结构。两种最常使用的两词搭配模式是“形容词 + 名词”和“名词 + 名词”(后者称为名-名复合词)。表 1.5 显示了这样的二元组的结果,这个结果是我们从语料库中抽取出来的“形容词 + 名词”和“名词 + 名词”搭配对。所有的结果似乎都是一个词组,并且可以成为词典的条目,也有一些例外,如 *last year* 和 *next year*。

我们对“搭配识别”的简短研究表明,在统计自然语言处理中,建模和数据分析之间是相互作用的,需要反复衡量。在上面的研究中,我们最初的模型把搭配简单地认为是频繁出现的二元组。但是,在分析了这个模型得到的结果之后,发现了问题并且提出了一个精炼的模型(这个模型认为搭配 = 带有特殊词性标注模型的频繁出现的二元组)。这个模型还需要进一步精简,因为像 *next year* 这样的二元组也被不正确地识别为搭配了。但是,目前我们暂不进行关于搭配的研究,在第 5 章将继续这方面的论述。

表 1.4 纽约时报中最常用的二元组

频率	单词1	单词2
80 871	of	the
58 841	in	the
26 430	to	the
21 842	on	the
21 839	for	the
18 568	and	the
16 121	that	the
15 630	at	the
15 494	to	be
13 899	in	a
13 689	of	a
13 361	by	the
13 183	with	the
12 622	from	the
11 425	New	York
10 007	he	said
9775	as	a
9231	is	a
8753	has	been
8573	for	a

表 1.5 过滤后最频繁的二元组。应用词性过滤器之后,纽约时报中最频繁的二元组

频率	单词1	单词2	词性组合模式
11 487	New	York	A N
7261	United	States	A N
5412	Los	Angeles	N N
3301	last	year	A N
3191	Saudi	Arabia	N N
2699	last	week	A N
2514	vice	president	A N
2378	Persian	Gulf	A N
2161	San	Francisco	N N
2106	President	Bush	N N
2001	Middle	East	A N
1942	Saddam	Hussein	N N
1867	Soviet	Union	A N
1850	White	House	A N
1633	United	Nations	A N
1337	York	City	N N
1328	oil	prices	N N
1210	next	year	A N
1074	chief	executive	A N
1073	real	estate	A N

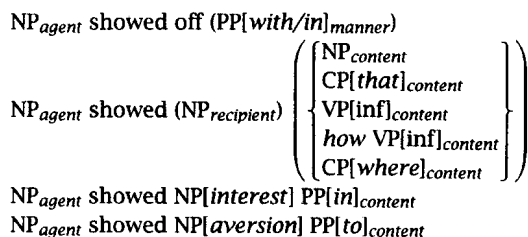
1.4.5 词汇索引

假设我们对含有动词的句法框架感兴趣,现在来进行最后一个数据探测。人们已经研究过了怎样使计算机自动找到这些框架,但我们也仅仅是把计算机作为一个工具来寻找合适的的数据。为了寻找动词框架,人们通常使用上下文关键词(KWIC)索引程序来产生数据表示,就像图 1.3 中的一样。在这样的表示中,所有出现的词汇都被列出,并且词的上下文环境也分别列在它的左右两边。通常,KWIC 程序允许按照左边或者右边的上下文对匹配词汇进行分类。然而,如果我们感兴趣的是语法框架,而不是具体的词汇,那么这种分类就会有限制。图中数据显示了词 *showed* 在小说“Tom Sawyer”中的出现情况。*showed off* 这种用法出现了 5 次(事实上它们都是在文章的一段中出现的),并且都是在双引号中,这可能是因为当时它是一种新的用法,或者可能是因为 Twain 认为这种表达是俚语。所有这些用法都是不及物的,虽然它们有些带有介词短语作为修饰语。与之相反的是,有 4 种及物动词用法,它们带有直接的宾语(6, 8, 11, 12),这些及物动词用法之间也存在着一些有趣的区别,例子 8 没有动词的行为主体,而例子 12 表示了一种显而易见的感觉。还有一种是双宾语用法,可以表示某人对某人显示了什么(16)。还有三个例子,*show* 的宾语名词短语被表示成 *that* 从句(13, 15),或者被表示成非限定式问句形式的补语从句(7)。另外还有一个例子含有限定式问句形式的补语从句(10),但是它的人称主语被省略了。在最后的两个例子中,宾语名词短语在介词的后面,这是最符合语言习惯用法的结构(9, 14): *show an aversion PP[to]* 和 *show an interest PP[in]*。但是我们注意到,虽然非常符合语言习惯,但是它们并不是完全僵化的形式,因为在这两种情况下宾语名词被修饰形成了更加复杂的名词短语。图 1.4 中总结了这些句法模式。

1	could find a target. The librarian	"showed	off" - running hither and thither w
2	elights in. The young lady teachers	"showed	off" - bending sweetly over pupils
3	ingly. The young gentlemen teachers	"showed	off" with small scoldings and other
4	seeming vexation). The little girls	"showed	off" in various ways, and the littl
5	n various ways, and the little boys	"showed	off" with such diligence that the a
6	t genuwyne?" Tom lifted his lip and	showed	the vacancy. "Well, all right," sai
7	is little finger for a pen. Then he	showed	Huckleberry how to make an H and an
8	ow's face was haggard, and his eyes	showed	the fear that was upon him. When he
9	not overlook the fact that Tom even	showed	a marked aversion to these inquests
10	own. Two or three glimmering lights	showed	where it lay, peacefully sleeping,
11	ird flash turned night into day and	showed	every little grass-blade, separate
12	that grew about their feet. And it	showed	three white, startled faces, too. A
13	he first thing his aunt said to him	showed	him that he had brought his sorrows
14	p from her lethargy of distress and	showed	good interest in the proceedings. S
15	ent a new burst of grief from Becky	showed	Tom that the thing in his mind had
16	shudder quiver all through him. He	showed	Huck the fragment of candle-wick pe

图 1.3 动词 *showed* 的上下文关键词(KWIC)表示

收集这些关于动词出现模式的信息,不仅可以作为外语学习者的词典数据,而且还可以用来指导统计句法分析。在统计自然语言处理的工作中,主要的任务就是对大规模数据(如词汇索引和搭配候选列表)进行统计。在一个语言工程开始的时候,这些工作有助于理解重要的语言现象,然后可以精炼最初的模型,并且在最后评价得到的结果。

图 1.4 “Tom Sawyer”中动词 *showed* 的句法框架

1.5 深入阅读

Chomsky(1965:47ff, 1980:234ff, 1986)讨论了语言处理中理性主义方法和经验主义方法之间的区别,并且给出了对理性主义方法的定位。近来经验主义者针对这些论点的比较详细的回应是(Sampson 1997)。对于那些曾经有产生式(计算式)语言学背景的人来说,他们很疑惑统计自然语言处理能为他们做什么,并且不知道怎样把它联系到他们以前关心的问题上,阅读 Abney(1996b)将是一个很好的起点。为了能够自举归纳,观测数据在某种一般化当中必然含有某些倾向性,这种现象在机器学习的文献中已经被 Mitchell(1980)指出了,他把这种倾向性称为统计倾向性偏置(bias)。在传统英语语料库语言学的一些棘手问题中,Firth的工作有很大的影响,这些工作完全包括在了 Stubbs(1996)中。统计自然语言处理团体多半起源于 AT&T 的工作,可以看到的一个例证是(Church and Mercer 1993:1)。Halliday 的方法应用于语言研究是 Halliday(1994)提出的。

语言学中语法合理性判断的完整讨论在 Schütze(1996)和 Cowart(1997)中可以找到。Cowart 认为可以使用说话人数量的判断标准,这和本书所提倡的方法是相通的,并且他反对 Chomsky 探求说话人本身具有语法功能的方法。对于这本著作在感性认识方面的一个好的切入点是(Harnad 1987)。

Lauer (1995b:ch.3)提倡的方法包括了语义的概率分布。参见第 12 章的深入阅读,文中涉及了其他统计自然语言处理的工作,包括语义的映射表示。

有关 *kind of* 和 *sort of* 的讨论是以 Tabor(1994)的工作为基础的,它是一种可以用来引证的资源。Tabor 提供了一个连接模型,显示了使用频率的改变怎样引起句法性质的改变。关于句法性质逐渐改变的大量研究可以在 Hopper and Traugott(1993)中找到,这是一本关于语法合理性的书籍。

Anderson(1983, 1990)和 Suppes(1984)支持概率方法,认为在认知中概率机制扮演了重要的角色。在(Oaksford and Chater 1998)中有关于这个方面的记述,描述了不同的认知体系结构,包括关联机制。语言是认知现象的最好诠释,这是认知语言学家的主要观点,参见(Lakoff 1987)和(Langacker 1987, 1991)。但是,由于认知语言学的形式化问题,很多认知语言学家并不认同概率理论。也可参见 Schütze(1997)。

小说“Tom Sawyer”在 Internet 上可以免费使用,最近可以从弗吉尼亚电子文本中心(Virginia Electronic Text Center)或其他资源获取(读者可以浏览相关网页)。

Zipf 的工作开始于(Zipf 1929),这是他的博士论文。他的两本主要著作是(Zipf 1935)和

(Zipf 1949)。有趣的是,我们注意到和 Zipf 同时代的语言学家对他的观点提出了苛刻的批评,例如,读者可以参阅(Kent 1930)和(Prokosch 1933)。在这些批评中有一部分集中在了 Zipf 的一些夸张的叙述上,(Kent 1930:80)指出:语音学和词法上的问题不可能被一个主要的概括性的公式全部解决),即使这样,这些批评也反映出人们在语言学中使用概率方法的一种复杂的心情。然而,像 Martin Joos 和 Morris Swadesh 这样杰出的美国语言结构学家,已经开始为统计语言研究收集资料了,Joos(1936)强调,是否在语言学中使用统计方法,这个问题应该和 Zipf 宣称的特殊观点分离开来进行评价。

另外还有(Mandelbrot 1954)。Mandelbrot 对 Zipf 法则的研究主要集中在(Mandelbrot 1983),要特别关注的是第 38 章、第 40 章和第 42 章。Mandelbrot 把他一生的工作方向(导致他著名的对碎片集和 Mandelbrot 集的研究成果)归因于阅读了(Zipf 1949)的评述。

由于重要的文学工作和宗教工作的需要,词汇索引最初是手工构建的。出于分类和索引文章标题及摘要的目的,计算机索引开始于 20 世纪 50 年代末期。Luhn(1960)开发了第一个计算机索引程序,并且发明了术语 KWIC。

1.6 习题

1.1 [★★ 需要一些语言学的知识]试着想像语言中的其他一些非绝对现象(noncategorical phenomena),它们似乎和语言的演化有关系。开始时,我们可以考虑下面的句子对,并试着找到它们的问题。可以简单地认为这个词属于两个范畴,或者认为这个词属于混合范畴,这样做是否可以解决下面的问题?

- a. On the weekend the children had *fun*.
b. That's the *funnest* thing we've done all holidays. (1.19)

- a. Do you get much *email* at work?
b. This morning I had *emails* from five clients, all complaining. (1.20)

1.2 [★★ 最好在阅读第 4 章之后再答此题]在另外的文本中重复 1.4 节的一些结果。作为一种选择,你可以使用和我们一样的文章,这样就可以很方便地检查自己的结果。在这种情况下,期望的结果和我们的类似,因为准确的统计数值要依赖于各种细节问题,比如什么是词汇,不同的情况如何区别,等等。

1.3 [★]证明:在 $B = 1$ 且 $\rho = 0$ 时, Mandelbrot 法则可以简化为 Zipf 法则。

1.4 [★★]使用 1.4.3 节中描述的随机字母生成器,构造一个类似表 1.3 的表(这个构造器产生从 a 到 z 的字母和空格,这些字符都有相同的生成概率 $1/27$)。

1.5 [★★]思考一些识别搭配的方法,这些方法可能会比本章中使用的方法更好。

1.6 [★★]如果做了上面的练习,试着分析所用的方法,看看为什么会有这样的效果。

1.7 [★★]编写一个程序,使其可以从一个文本文件中生成 KWIC 表示,并允许使用者选择感兴趣的词汇和上下文环境的大小。

第2章 数学基础

在下一章引入语言学基本知识之前,我们首先介绍在统计自然语言处理中所涉及到的统计学和信息论方面的基本概念。事实上,对任何在统计语言学领域从事基础性研究的学者来说,了解概率论、统计学、信息论以及语言学的知识都是很有必要的。本章不可能对其中各个领域的知识都做详尽的介绍和讨论,但是我们将尽可能清楚地阐述本书内容中所涉及的数学知识。这里假设大家在阅读本章内容之前已经具备基本的数学基础,并大概了解一般的数学符号表示法和数学技巧。这些数学基础包括了集合论、函数关系、多项式、向量、代数等方面的最基本的概念。关于本书所用的数学符号,读者可以参见所附的符号表。

如果你对本章所涉及的数学内容都很熟悉,可以跳过本章直接阅读后续的章节。但是,如果你对其中某个数学领域的知识不甚了解,我们还是建议你仔细阅读整章的内容。基于上面所阐述的种种原因,本章的重点将在于阐明数学理论方法,而不是其具体如何应用。

2.1 概率论基础

为了方便读者理解本书内容,这一节将书中涉及到的概率论基础知识要点给出一个提纲挈领的阐述。

2.1.1 概率空间

概率论是预测事件发生的可能性大小的理论。例如,向上抛3个硬币,正面同时朝上的可能性有多大?虽然本书的最终目的是自然语言处理的应用,但是以硬币或者骰子为例,会更有利于理解数学概念,因为它们很简单而且很直观。

一般来说,通过实验观察的方法可以刻画某一事件发生的可能性,比如抛硬币(假设硬币质地均匀)。但是,最关键的是如何定义我们所做的随机实验。所谓的随机实验是指其结果事先不能完全确定的实验。我们称 Ω 为一个样本空间或者基本事件空间,它是一个样本点或者基本事件的集合。样本空间 Ω 既可以是离散的(至多无限可数个样本点),也可以是连续的(不可数个样本点,比如测量身高时,身高取样为连续的值)。对于自然语言处理中的具体问题,主要是离散样本空间,其中包含有限个样本。设定 A 为样本空间 Ω 的一个子集,在上面的抛硬币实验中,其中一次的结果可能是一个硬币正面朝上,另外两个朝下,这个结果可以看做一次取样。值得注意的是,样本空间 Ω 是所有可能实验结果的集合,而空集则是不可能的事件的集合 $\emptyset$ 。概率论的基础在于事件集合 $\mathcal{F}$ 其实形成了一个 σ -field(包含一个最大元素 Ω 的集合,并且该集合对于它的元素求补运算和有限并集运算封闭)。上面的这些要求对于样本空间的幂集,即事件空间(event space)是很容易满足的(即,样本空间中所有子集的集合,通常记为 2^Ω)。

我们知道,概率值是介于0和1之间的一个值,0表示不可能发生,而1则表示必然事件。

所谓概率函数就是要在整个样本空间 Ω 分配概率值, 概率值总和为 1。一个离散概率函数的形式为 $P: \mathcal{F} \rightarrow [0, 1]$, 满足如下两个条件:

- $P(\Omega) = 1$
- 可加性: 对于不相交的集合 $A_j \in \mathcal{F}$ (例如, $A_j \cap A_k = \emptyset, j \neq k$)

$$P\left(\bigcup_{j=1}^{\infty} A_j\right) = \sum_{j=1}^{\infty} P(A_j) \quad (2.1)$$

其中 $P(A)$ 为事件 A 的概率, 定义说明了概率可加性的性质, 比如 3 个无关事件发生的概率值等于每个事件概率值的和。而且, 我们知道一个实验必然存在一些基本事件作为它的结果, 所以概率和为 1。运用最基本的集合论的知识, 可以从中推导出概率函数的更多性质 (见习题 2.1)。

一个完备的概率空间应该由样本空间 Ω 、概率函数 P 、事件的 σ -field 这 3 部分组成。在统计自然语言处理中, 我们的目标就是为建立的模型定义一个符合上述条件的概率空间。

例 1 一个质地均匀的硬币连续向上抛 3 次, 试问两次都是正面朝上的概率值?

解答:

从中我们可以很清楚地看出实验的样本空间为:

$$\Omega = \{HHH, HHT, HTH, HTT, THH, THT, TTH, TTT\}$$

样本空间 Ω 中的可能结果是等概率出现的, 都为 $1/8$, 这种分布在概率论上称为均匀分布。在有限样本空间的条件下, 如果满足等概率性质, 那么 $P(A) = \frac{|A|}{|\Omega|}$, ($|A|$ 表示集合 A 中的元素数目), 而正面朝上的事件集合表示为:

$$A = \{HHT, HTH, THH\}$$

因此

$$P(A) = \frac{|A|}{|\Omega|} = \frac{3}{8}$$

2.1.2 条件概率和独立性

我们有时可能具备对实验的某种先验知识, 从而可以判断哪些事件更可能发生。条件概率所表达的正是这样一种概念。不考虑已知知识对概率值的影响, 原有的概率值称为事件的先验概率 (prior probability), 而加入已知知识后, 原有的概率值将发生变化, 称为事件的后验概率 (posterior probability)。我们仍然以例 1 抛硬币为例, 如果已知第一次硬币正面朝上, 其余有 4 个可能的结果, 其中有两个是正面向上, 那么两个正面朝上这一事件发生的概率为 $1/2$ 。假设事件 B 的概率已知, 那么事件 A 发生的条件概率为 ($P(B) > 0$):

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \quad (2.2)$$

当 $P(B) = 0$ 时, 我们有:

$$P(A \cap B) = P(B)P(A|B) = P(A)P(B|A) \quad [\text{乘法律}] \quad (2.3)$$

可以采用上面公式中的任何一个,因为集合的交运算满足对称性($A \cap B = B \cap A$)。图 2.1 给出了图形解释。

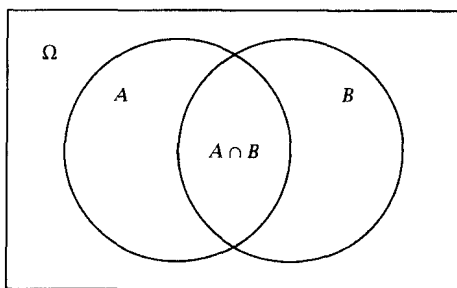


图 2.1 条件概率 $P(A|B)$ 的计算图示,其计算公式为 $P(A \cap B)/P(B)$

将式(2.3)推广到多事件的情况,用公式表示如下:

$$P(A_1 \cap \dots \cap A_n) = P(A_1)P(A_2|A_1)P(A_3|A_1 \cap A_2) \cdots P(A_n|\cap_{i=1}^{n-1} A_i) \quad (2.4)$$

▼ 在统计自然语言处理中,上面这个链式规则很有用处,比如第 9 章中用于推导马尔可夫模型的性质。

两个事件 A 和 B 互相独立当且仅当它们满足 $P(A \cap B) = P(A)P(B)$,除非 $P(B) = 0$ 。也相当于 $P(A) = P(A|B)$ (即事件 B 不影响事件 A 发生的概率);否则事件 A 和 B 为相关的事件。另外,当事件 A, B, C 满足 $P(A \cap B|C) = P(A|C)P(B|C)$ 时,称为 A 和 B 满足条件独立性。

2.1.3 贝叶斯定理

贝叶斯定理的优势在于计算概率时可交换相关事件的次序,也就是说,可以通过 $P(B|A)$ 的值间接计算 $P(A|B)$ 的值。尤其是当其中一个概率值很难直接计算时,可以利用另外一个概率值来计算未知的概率值。此法在本书中应用很广泛,可以由式(2.2)和式(2.3)中给出的条件概率公式和链式规则直接推理得到:

$$P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(A|B)P(B)}{P(A)} \quad (2.5)$$

公式右边的分母 $P(A)$ 可以看做归一化常数,以保证其满足概率函数的性质。如果我们感兴趣的仅仅是事件发生的相对可能性,这时完全可以忽略这个分母,因为公式中的分母大小都是相同的。

$$\arg \max_B \frac{P(A|B)P(B)}{P(A)} = \arg \max_B P(A|B)P(B) \quad (2.6)$$

但是,我们也可以通过下式计算分母的大小。

$$\begin{aligned} P(A \cap B) &= P(A|B)P(B) \\ P(A \cap \bar{B}) &= P(A|\bar{B})P(\bar{B}) \end{aligned}$$

所以

$$\begin{aligned} P(A) &= P(A \cap B) + P(A \cap \bar{B}) && [\text{加法律}] \\ &= P(A|B)P(B) + P(A|\bar{B})P(\bar{B}) \end{aligned}$$

其中, B 和 $\bar{B}$ 将 A 集合分成两个不相交的非空的部分, 分别对其计算条件概率, 然后利用可加性的性质, 得到以上结果。现在将其推广之, 假设一组互不相交的集合 B_i 对 A 进行划分, 满足 $A \subseteq \bigcup_i B_i$, 则有:

$$P(A) = \sum_i P(A|B_i)P(B_i) \quad (2.7)$$

依照此式, 我们给出贝叶斯定理的更正式的数学定义:

贝叶斯定理: 假设 $A \subseteq \bigcup_{i=1}^n B_i$, $P(A) > 0$, 并且对于任何的 $i \neq j$, 满足 $B_i \cap B_j = \emptyset$, 则有:

$$P(B_j|A) = \frac{P(A|B_j)P(B_j)}{P(A)} = \frac{P(A|B_j)P(B_j)}{\sum_{i=1}^n P(A|B_i)P(B_i)} \quad (2.8)$$

例 2 假设我们对某种特殊的句法构造感兴趣, 比如英文中的寄生间隙 (parasitic gap) 结构, 它可能在 100 000 个句子中平均只出现一次, 语言学家 Joe 构造了一种复杂的模式匹配器来识别这类句法结构。它的识别性能很好, 但不是最好的。当通过匹配判断某个句子内包含一个寄生间隙时, 判断正确的可能性为 0.95; 反之, 判断错误的概率为一个很小的概率值 0.005。那么, 当通过测试表明句子含有一个寄生间隙时, 通过匹配识别出这个结构的概率是多大?

解答:

设 G 表示所有含有该类特殊句法结构的事件集合, T 表示所有验证为真的事件集合, 那么我们有:

$$\begin{aligned} P(G|T) &= \frac{P(T|G)P(G)}{P(T|G)P(G) + P(T|\bar{G})P(\bar{G})} \\ &= \frac{0.95 \times 0.000\ 01}{0.95 \times 0.000\ 01 + 0.005 \times 0.999\ 99} \approx 0.002 \end{aligned}$$

这里, 我们在公式的分母中对于是否存在这种结构对事件空间进行了划分, 虽然语言学家 Joe 的实验看起来是完全可信赖的, 但是我们发现它并不能给予我们所期望的帮助。测试表明, 在验证为真的句子中, 平均每 500 句只有一个句子包含这种匹配句法结构。之所以会出现这种现象, 是因为包含这种特殊句法结构的句子的先验概率很小。

▼ 在稍后将要介绍的 2.2.4 节中, 贝叶斯定理是噪声信道模型的核心内容。

2.1.4 随机变量

随机变量 X 是一个简单的样本空间到实数域集合的映射函数, $X: \Omega \rightarrow \mathbb{R}^n$, n 值常取 1, 其中 $\mathbb{R}$ 表示实数域集合。随机变量用于讨论和事件空间相关的数值的概率, 相当于考虑一个抽象的随机过程, 它按照一定的概率大小生成任意数字, 离散随机变量是这样一个函数: $X: \Omega \rightarrow S$, 其中 S 为实数集合的有限可数子集 $\mathbb{R}$, 如果满足 $X: \Omega \rightarrow \{0, 1\}$, 则 X 称为一个指示随机变量。

例 3 假设我们掷两个骰子, 骰子的点数和就可以作为一个随机变量: $S = \{2, \dots, 12\}$, 如图 2.2 所示。

随机变量有一个取值的范围,因此我们避免了直接和事件本身打交道,而是处理代表它们的随机变量的值,这样就转化为数学上的问题,方便了处理过程。同时,为了更清楚地描述随机变量的特性,我们引入概率密度函数(probability mass function,简称 pmf)的概念,它表示随机变量取不同值的概率:

$$\text{概率密度函数 } p(x) = p(X = x) = P(A_x) \quad \text{其中, } A_x = \{\omega \in \Omega : X(\omega) = x\} \quad (2.9)$$

第一个骰子	第二个骰子											
	1	2	3	4	5	6						
6	7	8	9	10	11	12						
5	6	7	8	9	10	11						
4	5	6	7	8	9	10						
3	4	5	6	7	8	9						
2	3	4	5	6	7	8						
1	2	3	4	5	6	7						
x	2	3	4	5	6	7	8	9	10	11	12	
$p(X = x)$	$\frac{1}{36}$	$\frac{1}{18}$	$\frac{1}{12}$	$\frac{1}{9}$	$\frac{5}{36}$	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{1}{9}$	$\frac{1}{12}$	$\frac{1}{18}$	$\frac{1}{36}$	

图 2.2 表示两个骰子点数和的随机变量,表中列出了已知样本点条件下的随机变量 X 的数值大小,最下面的两行给出了 X 的概率密度函数 $p(x)$

一般采用小写罗马字符表示概率密度函数,如果随机变量 X 是按概率密度函数 $p(x)$ 分布的,则记为: $X \sim p(x)$ 。

注意,只有有限可数个点满足 $p(x) > 0$ 的性质,还存在一个样本集合 $\{x_i : i \in \mathbb{N}\}$ 满足 $p(x) = 0$ 。对于一个离散的随机变量 X ,满足下面这个等式:

$$\sum_i p(x_i) = \sum_i P(A_{x_i}) = P(\Omega) = 1$$

反之亦然,任何满足这些约束的函数都可以认为是一个概率密度函数。

▼ 随机变量的概念将贯穿于稍后在 2.2 节对信息论的介绍中。

2.1.5 期望值和方差

期望值是一个随机变量的各个取值的平均值,设 X 为随机变量,其概率密度函数为 $p(x)$,满足 $\sum_x |x| p(x) < \infty$,则期望值为:

$$E(X) = \sum_x x p(x) \quad (2.10)$$

例 4 设随机变量 Y 表示一个骰子面上的点数,假设骰子质地均匀,即每一面出现的概率都为 $1/6$,则它的期望值为:

$$E(Y) = \sum_{y=1}^6 y p(y) = \frac{1}{6} \sum_{y=1}^6 y = \frac{21}{6} = 3\frac{1}{2}$$

这个期望值就表示多次掷骰子实验得到的总点数和除以实验总次数。

如果 $Y \sim p(y)$ 表示一个随机变量,那么函数 $g(Y)$ 可以定义为一个新的随机变量。如果 $E(g(Y))$ 有定义,则有:

$$E(g(Y)) = \sum_y g(y) p(y) \quad (2.11)$$

例如,如果 $g(Y) = aY + b$ 为一个线性函数,那么 $E(g(Y)) = aE(Y) + b$,我们一般有 $E(X + Y) = E(X) + E(Y)$,如果随机变量 X 和 Y 相互独立,则还满足 $E(XY) = E(X)E(Y)$ 。

一个随机变量的方差描述该随机变量的值偏离其期望值的程度。随机变量的方差为:

$$\begin{aligned}\text{Var}(X) &= E((X - E(X))^2) \\ &= E(X^2) - E^2(X)\end{aligned}\quad (2.12)$$

我们通常提到的标准方差是方差的平方根,对于任意样本数据集合,用符号 μ 表示样本均值,用 σ 表示标准方差,用 σ^2 表示方差。

例 5 计算例 3 中表示两个骰子点数和的随机变量的期望值和方差。

解答:

利用例 4 中得到的期望值以及式(2.11),可以得到:

$$E(X) = E(Y + Y) = E(Y) + E(Y) = 3\frac{1}{2} + 3\frac{1}{2} = 7$$

方差为:

$$\text{Var}(X) = E((X - E(X))^2) = \sum_x p(x)(x - E(X))^2 = 5\frac{5}{6}$$

可以设想,如果是 11 面骰子,那么该随机变量的分布应该是均匀分布,其取值范围为 $\{2, 12\}$,计算可知它的方差大小为 10;本例中两次抛骰子的点数集中在 7,所以它的方差小于 11 面骰子的情况。

▼ 计算变量的期望值在信息论中是很重要的(见 2.2 节),5.2 节将介绍方差在统计自然语言处理中的应用。

2.1.6 符号表示

当阅读本书时,有注意大写 P 和小写 p 之间的差别。 P 一般用来表示概率函数, p 表示随机变量的概率密度函数。但是,符号 $P(\cdot)$ 和 $p(\cdot)$ 并不总是表示同一个函数。当我们提及不同的概率空间时,实际上就是在谈论不同的概率函数,即概率函数和具体的概率空间有关。通常用加下标的方法可以区分我们所讨论的具体问题。有时候用大写 P 表示几个函数,这就需要通过联系上下文以及随机变量参数的名称去辨别其含义。同一个等式可能表示几个不同的概率函数,认识到这一点很重要,即它们都用 P 表示。

2.1.7 联合分布和条件分布

我们用联合概率分布来描述样本空间中多个随机变量的分布,设两个离散随机变量 X 和 Y ,它们的联合密度函数可写为:

$$p(x, y) = P(X = x, Y = y)$$

而描述其中单个随机变量的概率密度函数称为边缘概率密度函数(marginal pmf),上面的两个离散随机变量的边缘概率密度函数可写为:

$$p_X(x) = \sum_y p(x, y) \quad p_Y(y) = \sum_x p(x, y)$$

一般条件下随机变量的边缘概率密度函数不能决定其联合密度函数,除非各个随机变量互相独

立。如果两个离散随机变量 X 和 Y 的分布独立,则它们的联合密度函数 $p(x, y) = p_X(x)p_Y(y)$ 。仍然以掷骰子为例,表示两个骰子点数的随机变量 X 和 Y 显然是互相独立的,所以两个骰子点数都为 6 的概率可按如下方式计算:

$$p(Y = 6, Z = 6) = p(Y = 6)p(Z = 6) = \frac{1}{6} \times \frac{1}{6} = \frac{1}{36}$$

对于联合概率分布和相交事件,有类似的结果,所以我们可以用联合分布来定义条件概率密度公式并推导出连锁计算公式:

$$\text{对于 } y, \text{ 有 } p_{X|Y}(x|y) = \frac{p(x, y)}{p_Y(y)} \quad \text{所以 } p_Y(y) > 0$$

$$p(w, x, y, z) = p(w)p(x|w)p(y|w, x)p(z|w, x, y)$$

2.1.8 概率函数 P 的估计

到目前为止,我们一直假设概率函数 P 为已知的,并举了几个简单的例子说明这个问题。但是在自然语言处理中如何定义呢?例如,英文句子 *the cow chewed its cud* 在日常生活中出现的概率是多大?这都是下面将要讨论的问题。此类语言事件的概率函数 P 在通常情况下是未知的,这就意味着我们必须去估计函数 P 。某一类结果在总的结果集合中所占的比例称为相对频率(relative frequency)。设 $C(u)$ 表示结果 u 在 N 次独立实验中出现的次数,其相对频率 $f_u = C(u)/N$,如果我们所做的实验次数很多,那么相对频率将趋向于一个稳定的值,这个性质为我们估计概率函数提供了一个必要的前提基础。

本书在以后的章节中将陆续介绍一些估计方法,尤其是在第 6 章中将进行比较详细的介绍。其中最常用的技巧就是假设某些语言“现象”服从某一类已知的概率分布,例如二项分布或者正态分布,这两种分布在统计学中已经有了广泛的研究,下一节将做具体的介绍。我们称这种方法为参数化方法,它有两个优点。这意味着我们有一个数据生成过程的概率模型,要在一系列可能的概率分布中确定出一个来;另一方面,因为只有几个参数需要决定,它需要的训练样本数据的规模也不会很大,我们甚至可以计算出为达到理想的估计结果所需的训练数据量。

但是,预定义的概率模型并不能描述所有的自然语言问题,例如新闻报纸中涉及某类主题的文章中的词的分布。这种分布是不规则的,所以不能简单地采用一些模型(例如二项分布)来刻画它,否则将会遇到很多问题,得出的概率估计也是错误的。

这种情况下一般都使用非参数化的方法(non-parametric)或者称为自由分布的方法(distribution-free),即用简单的统计方法去估计概率分布,估计的结果相当于产生了一个离散的概率分布。当然,通过插值法也可以得到一个连续的分布函数 P (概率密度函数必须满足是平滑函数的条件)。非参数化的方法存在两个缺陷。第一,出于训练数据有限的原因,估计的结果往往还需要平滑操作。为了进行平滑,往往又利用了假设的隐含分布,相当于又回到了参数化方法的老路上来;第二,采用非参数化的方法等同于我们对样本数据如何生成一无所知,因此先验知识的匮乏意味着我们不得不需要更多的训练数据去估计参数。

- ▼ 非参数化的方法常常用于自动分类问题,尤其是当样本数据隐含分布未知时,最近邻算法就是这类方法的一个典型例子,我们将在第 16 章对文本分类的介绍中详细讨论这一问题。

2.1.9 标准分布

在实际应用中使用的往往是某一个特定的概率密度函数,因为经常会发现所用的概率分布函数只是一个一般性公式取不同的常数参数而已。也就是说,它只是该分布族函数中的一个特例,是将分布族中的参数取不同值的结果。所以,当我们谈论某一个特定分布函数时,是其相应的分布族参数取常数值的结果。在书写分布的参数时,我们通常将它和一般的随机变量用冒号(;)隔开。本节选取二项分布和正态分布这两个分布做简短的介绍。

离散分布函数:二项分布(binomial distribution)

当重复一个只有两种输出(假设为 0 和 1)的实验,比如柏努利实验,并且各次实验之间相互独立时,我们就说实验的结果服从二项分布,前述抛硬币的实验是一个典型的例子。而在自然语言处理中,语料库的当前句子和前面句子之间肯定不是完全互相独立的,所以我们应用时只能近似认为它们服从二项分布,但是在有些条件下,比如词之间的依赖程度很小时,我们可以认为它们的分布是互相独立的。当我们对某个样本事件进行计数,以表明某事务存在与否或某属性存在与否时,往往忽略了两次实验之间的依赖性,这相当于隐含地引入了一个二项分布。二项分布在统计自然语言处理中应用得很广泛,例如估计英文文本中含有单词 *the* 的句子所占的百分比,或者确定一个动词在语言中是否常被用做及物或者非及物动词。

对于二项分布族函数,设成功的概率为 p ,那么 n 次独立实验中成功的次数为 r 的概率为:

$$b(r; n, p) = \binom{n}{r} p^r (1-p)^{n-r} \quad \text{其中, } \binom{n}{r} = \frac{n!}{(n-r)!r!} \quad 0 \leq r \leq n \quad (2.13)$$

$\binom{n}{r}$ 表示在不考虑实验次序的情况下从 n 个物件中选择 r 个物件的不同组合的数目。图 2.3 中给出了一些二项分布的例子,二项分布的期望值为 np ,方差为 $np(1-p)$ 。

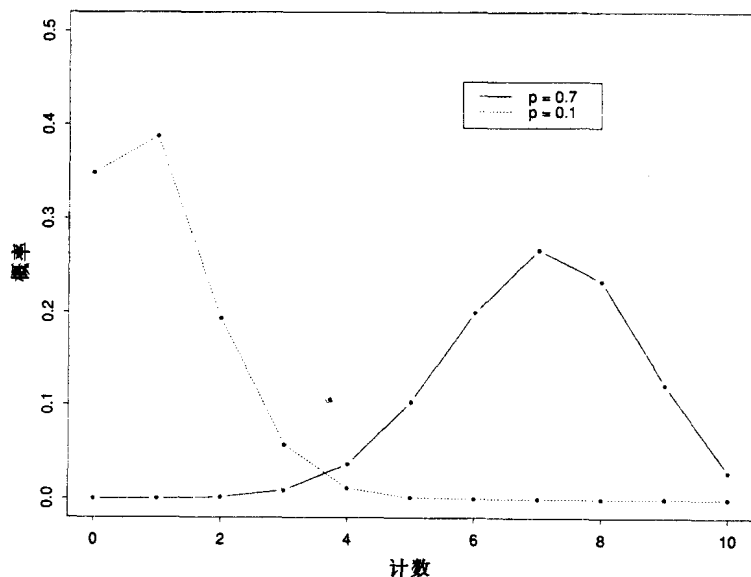


图 2.3 两个二项分布的例子: $b(r; 10, 0.7)$ 和 $b(r; 10, 0.1)$

例6 设 R 表示 n 次抛硬币实验中正面朝上的次数,假设硬币质地均匀,正面朝上的概率为 p 。

我们计算二项分布的公式如下:

$$p(R=r) = b(r; n, p)$$

在每个 n 次抛硬币实验中, r 次正面朝上, $(n-r)$ 次正面朝下的基本事件的概率为

$h^r(1-h)^{n-r}$, 总计有 $\binom{n}{r}$ 个这样的结果。

- ▼ 本书的很多章节都涉及二项分布问题,例如第6章中的 n -gram 问题,8.2节中的假设检验问题。当实验有两个以上的结果时,二项分布问题就转化为多项式分布(multinomial distribution)。
- ▼ 第6章中的零阶的 n 阶语言模型是一个简单的多项式分布的例子。
- ▼ 本书中涉及到的另外一个离散分布是泊松分布,我们将在15.3.1节讨论有关内容。5.3节讨论了柏努利分布,实际上它是二项分布的一个特例,取实验次数等于1的情况,即等同于 $b(r; 1, p)$ 。

连续分布函数:正态分布

到目前为止,我们讨论的都是离散概率分布和离散随机变量,但是很多物体对象,例如高度或者长度的测量值在实数域上都是连续的,必须用连续随机变量描述。连续随机变量的取值在某个区间 (a, b) 。读者可以直观地理解,和离散随机变量相比,相当于把点变为区间,求和变为积分,下面我们简单介绍一种连续分布:正态分布。

如前所述,当描述世界上的很多事物时,比如人的身高或者智商,我们将得到类似报纸传媒中经常提到的钟形曲线,在统计学中称之为正态分布。图2.4中给出了一个正态分布曲线图,图中的概率密度函数并不能直接给出概率值的大小(实际上在连续分布函数中某一点的概率为0)。在某个区间的概率大小由 x 轴和函数曲线所界定的面积决定。

正态分布函数有两个参数,即均值 μ 和标准方差 σ ,其函数公式为:

$$n(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)} \quad (2.14)$$

满足参数 $\mu=0$, $\sigma=1$ 的正态分布称为标准正态分布,附录A中给出了部分标准正态分布的参数表。

虽然从某种程度上讲,“正态分布”这个名称比“钟形曲线”更合适一些,但是在统计自然语言处理和模式识别领域中更多地称之为高斯分布^①。

在许多统计应用中,我们用连续的正态分布函数来近似离散的二项分布,通过比较图2.3和图2.4,可以发现两者的函数曲线表面上看起来就有形似之处,当基本事件的发生概率满足一定大小的限制或者样本数据集合很大时(一般有 $np(1-p) > 5$),这种近似是可以接受的。但是,在自然语言处理中存在一些稀有事件。例如,设事件为短语 *shade tree mechanics* 在文本中的出现次数,显然即使是在大规模的文本中,这个短语的出现频率也很小,这时如果用正态函数曲线取近似二项分布,那么结果会出现很大的误差。

^① 德国数学家高斯第一个引入了正态分布的概念。天文学家和天文观察者经常需要重复进行观测度量,高斯使用正态分布建立其错误率的模型。事实上,正态曲线是由法国数学家棣莫弗(Abraham de Moivre)首先发现的。

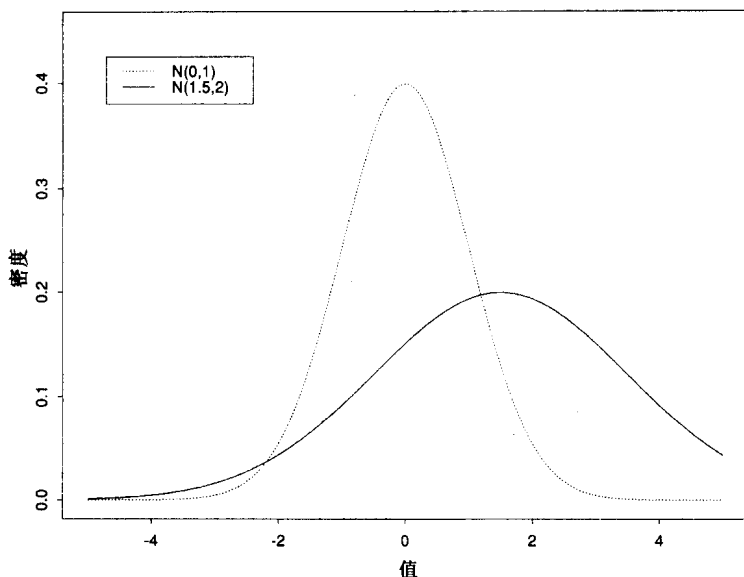


图 2.4 两个正态分布的曲线图: $n(x;0,1)$ 和 $n(x;1.5,2)$

- ▼ 高斯分布经常用于解决聚类问题,本书的第 14 章中将具体讨论这个问题。这里给出的仅仅是一维或单变量的正态分布,在第 14 章中我们将其扩展到高维情况。
- ▼ 本书中涉及到的其他一些连续分布函数包括双曲线分布(参见 1.4.3 节), t 分布用在假设检验上(参见 5.3 节)。

2.1.10 贝叶斯统计

到目前为止,我们简短介绍了频率统计学,但是并不是所有的人都同意传统统计学的哲学基础,其中一个持不同意见的重要学派就是贝叶斯学派。实际上,甚至贝叶斯学派内部也存在一些分歧,但是这不是本节内容的重点,我们仅仅希望简短地介绍贝叶斯的一般方法,因为这是在统计自然语言处理中经常采用的一种方法。

贝叶斯更新

仍然以抛硬币为例,假设我们重复 10 次,其中 8 次结果是正面朝上。从一个频度论者的角度来看,他会认为 10 次实验中 8 次正面朝上是一种极大似然估计(参见 6.2.1 节)。但是,假设一些人会检查硬币,并且发现硬币是质地均匀的,那么他可能不会接受这个结果,他会趋向于认为从长远的角度来看,硬币正反两个面朝上应该是等概率的事件,上面 10 次实验中 8 次正面朝上的事件仅仅是在小样本条件下取得的结果。换句话说,人们通常都有一种先验的知识,即使存在明显的事实和这种先验知识相悖。贝叶斯学派不赞同这种先验知识的合理性,他们会在不断地获取证据的基础上重复更新这些先验知识,更新方法是利用贝叶斯定理。

继续以抛硬币为例,我们用 μ_m 表示模型^①,令 $P(\text{head}) = m$,设 s 表示观察到的事件的特

① 这里的模型是指按照某种理论知识体系来解释世界上的一些规律性概念,一个概率模型中可能包含其分布函数和某些参数值的说明。我们在式(2.15)的表示符号中使用了一些偷懒的方法,前面曾经以事件空间的子集作为条件概率的后项,因此这里把模型直接写入了条件概率公式。

定序列,其中有 i 次正面朝上, j 次正面朝下。那么对于任意的 m , $0 \leq m \leq 1$,我们有:

$$P(s|\mu_m) = m^i(1-m)^j \quad (2.15)$$

从频度论者的角度看,我们的目的是寻找概率的一个极大似然估计:

$$\arg \max_m P(s|\mu_m)$$

所以,这个多项式就需要利用导数法寻找最大值,其结果和我们的直观感觉一样为 $i/(i+j)$, 对于这个例子则是 0.8。

如果某人设想硬币质地均匀,并预先假定了一个先验概率分布来验证他这个假设模型正确与否,很多人都认为其概率密度大部分都集中于 $1/2$ 附近,所以可能会设想一个类似高斯分布的模型去刻画它,但是为了求解的便利,我们用一个多项式来描述这类事件先验分布:

$$P(\mu_m) = 6m(1-m) \quad (2.16)$$

之所以选择这个多项式,一方面是因为该分布的中心点为 $1/2$,另外曲线在区间 $[0,1]$ 之间的面积为 1。

当观察到序列 s 之后,可能会改变原来所谓的先验知识,因为我们知道了更多的证据信息,通过式(2.15),式(2.16)以及利用贝叶斯定理可以验证这一点,计算如下:

$$\begin{aligned} P(\mu_m|s) &= \frac{P(s|\mu_m)P(\mu_m)}{P(s)} \\ &= \frac{m^i(1-m)^j \times 6m(1-m)}{P(s)} \\ &= \frac{6m^{i+1}(1-m)^{j+1}}{P(s)} \end{aligned} \quad (2.17)$$

现在我们假设 $P(s)$ 为 s 的先验概率,并且它的值独立于模型 μ_m ,因此在寻找最大化上面式子的 m 时可以忽略 $P(s)$ 的影响,然后我们对分子求导得到最大值。以抛硬币为例,10 次实验中有 8 次正面朝上,2 次正面朝下,按照这个值计算可得:

$$\arg \max_m P(\mu_m|s) = \frac{3}{4}$$

事实上,我们原来所做的是一个很弱的先验假设。多项式分布函数是一个以 $1/2$ 为中心的平坦曲线,所以导致得出的结果在一定程度上偏离了 $1/2$,但很重要的一点是毕竟没有得出前面的 0.8 的结果值。因此,如果我们假定一个更强的先验假设,也许可能使得出的后验概率值更逼近 $1/2$ (参见习题 2.8)。

但是我们如何处理公式中的分母 $P(s)$ 呢? 一个错误的观点认为:既然序列 s 对我们来说是已知的,那么完全可以认为其概率值为 1。但事实上并非如此,它是一个边缘概率,可以通过累加 $P(s|\mu_m)$ 加权值得到,和式(2.8)相同,对于权重为 μ_m 的概率值,有如下的积分公式:

$$\begin{aligned} P(s) &= \int_0^1 P(s|\mu_m)P(\mu_m)dm \\ &= \int_0^1 6m^{i+1}(1-m)^{j+1}dm \end{aligned} \quad (2.18)$$

这个公式实际上是 β 积分的一个特例,统计学家对这个分布进行了深入研究,所以我们通过查资料可以直接得到以下结果:

$$P(s) = \frac{6(i+1)!(j+1)!}{(i+j+3)!} \quad (2.19)$$

但是,很重要的一点是: $P(s)$ 在公式中仅仅是作为归一化因子,以保证式(2.17)中的 $P(\mu_m | s)$ 满足是概率函数的条件。

通常来说,当样本数据以序列的方式给出,那么认为它们之间的分布独立是合理的,每当出现新的证据数据时,通过最大化后验概率(又称为 MAP 概率)的方法重新更新原来的先验假设,这个过程称为贝叶斯更新。

贝叶斯决策定理

贝叶斯的方法还有另外一个好处,可以评估哪个模型或者模型族能更好地适应数据,解释数据。假设我们实际上不能直接看到抛硬币的结果,例如,它可能是间接通过别人观察后所报告的结果。而到目前为止,可能是这样一种情况:假设报告的实验结果真实地反映出一个质量不均匀硬币的实验结果。我们将这类假设作为一类模型族,称为理论 μ ,模型中有一个参数专门用来表示硬币的质量分布。现在假设存在另外一种可能,我们每次抛出的是两个而不是一个质地均匀的硬币,当两个硬币都是正面朝下的时,结果认为是正面朝下,否则认为结果是正面朝上,我们称之为理论 ν 。按照理论 ν 的假设,如果 s 中观察到有 i 次正面朝上, j 次正面朝下,那么有:

$$P(s|\nu) = \left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^j \quad (2.20)$$

注意,在这两个理论中,一个需要自由参数,表示类似硬币的均匀度概念,另外一个则不需要任何参数。我们不妨假设这两个理论是等可能的,即:

$$P(\mu) = P(\nu) = \frac{1}{2} \quad (2.21)$$

现在我们就可以在已知观测样本数据的条件下,利用贝叶斯定理判断两个理论中哪一个更有可能:

$$P(\mu|s) = \frac{P(s|\mu)P(\mu)}{P(s)} \quad P(\nu|s) = \frac{P(s|\nu)P(\nu)}{P(s)}$$

读者容易混淆的地方可能是这里采用的符号表示的改变,实际上公式中的 $P(s|\mu)$ 值等同于式(2.19)中的 $P(s)$,因为那时我们假定理论 μ_m 为真,目标是计算 m 的值,但是现在 $P(s)$ 是作为 s 的先验概率出现的,并且理论 μ 真假未知。一种方法是直接计算两个模型的似然比,消去分母 $P(s)$,由式(2.19),式(2.20)和式(2.21)可以得到:

$$\begin{aligned} \frac{P(\mu|s)}{P(\nu|s)} &= \frac{P(s|\mu)P(\mu)}{P(s|\nu)P(\nu)} \\ &= \frac{\frac{6(i+1)!(j+1)!}{(i+j+3)!}}{\left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^j} \end{aligned} \quad (2.22)$$

如果这个值大于 1,那么我们选择理论 μ ,否则选择理论 ν 。我们也可以对其取对数,根据它是否大于 0 做出判断。

我们可以计算正面朝上和正面朝下次数的不同组合情况下的似然比,表 2.1 列出了 10 次

和 20 次实验对应的似然比概率值。如果正面朝上的次数少,那么似然比大于 1,理论 μ 为真。因为其具有自由参数,所以它基本上和任何样本数据不发生冲突。另一方面,如果样本数据分布基本上符合两个质量均匀硬币的理论(正面朝上的次数远远大于正面朝下的次数),那么其似然比小于 1。当存在更多的可利用数据时,假设理论 ν 为真,那么其正面朝上的次数所占的比例应该更接近 3/4。考虑存在两个可供选择的理论,我们称这种通过似然比大小判定的方法为贝叶斯最优决策(Bayes optimal decision)。

表 2.1 两个理论假设之间的似然比。左边 3 栏表示 10 次实验的数据,右边 3 栏表示 20 次实验的数据

10次抛硬币实验的结果			20次抛硬币实验的结果		
正面朝上的次数	正面朝下的次数	似然比	正面朝上的次数	正面朝下的次数	似然比
0	10	4.03×10^4	0	20	1.30×10^{10}
1	9	2444.23	2	18	2.07×10^7
2	8	244.42	4	16	1.34×10^5
3	7	36.21	6	14	2307.06
4	6	7.54	8	12	87.89
5	5	2.16	10	10	6.89
6	4	0.84	12	8	1.09
7	3	0.45	14	6	0.35
8	2	0.36	16	4	0.25
9	1	0.37	18	2	0.48
10	0	0.68	20	0	3.74

- ▼ 当存在更多而不是两个理论假设时,我们两两比较并选择一个最有可能的假设,在 7.2.1 节的词义消歧问题中会讨论一般形式的贝叶斯决策理论,并给出一个具体的实例。

2.2 信息论基础

信息论是由香农(Claude Shannon)在 20 世纪 40 年代建立的理论体系,参见(Shannon 1948)。虽然香农研究的实际目的是如何破解第二次世界大战时的敌方密码,但他感兴趣的是如何在非理想化的通信信道中传输尽可能多的信息。对于任何类型的信息源和信息通道,香农希望在理论上能给出数据压缩率(通过熵 H 来表示,或者称为柯尔莫哥洛夫复杂度 K)和数据传输率的最高值。在香农之前,人们普遍认为传输速率和错误率成正比,速度越高,错误越多。但是香农证明只要数据传输速率低于信道容量 C ,即可获得我们期望的任意错误率。

2.2.1 熵

设 $p(x)$ 为随机离散变量 X 的概率密度函数, x 属于某个符号或者字符的离散集合 $\mathcal{X}$:

$$p(x) = P(X = x), \quad x \in \mathcal{X}$$

例如,如果我们抛两个硬币,并计算实验中正面朝上的硬币数,实际上就是把这个值当做随机变量: $p(0) = 1/4, p(1) = 1/2, p(2) = 1/4$ 。

熵(entropy)表示单个随机变量的不确定性的均值,随机变量的熵越大,它的不确定性越大,也就是说,能正确估计其值的概率越小。熵的计算公式如下:

$$\text{熵 } H(p) = H(X) = - \sum_{x \in X} p(x) \log_2 p(x) \quad (2.26)$$

熵值度量了随机变量所包含的信息量的大小,它的单位为比特,所以我们取以 2 为底的对数,本书的公式中如果没有特别指明,都假设 2 为对数的底数,同时假定 $0 \log 0 = 0$ 。

例 7 假设我们抛出一个具有 8 个面的骰子,那么实验结果的熵值可计算如下:

$$H(X) = - \sum_{i=1}^8 p(i) \log p(i) = - \sum_{i=1}^8 \frac{1}{8} \log \frac{1}{8} = - \log \frac{1}{8} = \log 8 = 3 \text{ 比特}$$

这个结果是我们所期望的。随机变量的熵可以看做传输该随机变量表示的基本事件所需的平均信息长度,也就是说,如果我们希望以最有效的编码方式传送结果,最好的方法是用 3 个比特表示掷骰子的结果。

1	2	3	4	5	6	7	8
001	010	011	100	101	110	111	000

传输每个实验结果的代价为 3 比特,并且不存在其他任何一个比这个代价还要低的编码方案。通常传输一个概率为 $p(i)$ 的信息的最优编码位数是大于 $\lceil -\log p(i) \rceil$ 的最小整数值。现在我们把对数前面的负号移入对数公式内部,熵的计算公式变为:

$$H(X) = \sum_{x \in X} p(x) \log \frac{1}{p(x)} \quad (2.27)$$

没有统计学知识的人通常会把该式看做普通的累加公式,即对每个 x 计算 $p(x) \log(1/p(x))$ 的和值,但这是一种错误的直观感觉,实际上这个公式表达的是一个加权求值的概念,权重就是随机变量 X 的每个取值的概率,所以按照统计学理论,该式实际上定义了一个期望值。

$$H(X) = E \left(\log \frac{1}{p(X)} \right) \quad (2.28)$$

例 8 简单的波利尼西亚语言^① 似乎只是一系列字符随机序列,其中各个字符的频率如下:

p	t	k	a	i	u
1/8	1/4	1/8	1/4	1/8	1/8

那么每个字符的熵为:

$$\begin{aligned} H(P) &= - \sum_{i \in \{p, t, k, a, i, u\}} P(i) \log P(i) \\ &= - \left[4 \times \frac{1}{8} \log \frac{1}{8} + 2 \times \frac{1}{4} \log \frac{1}{4} \right] \\ &= 2.5 \text{ 比特} \end{aligned}$$

这个结果表明,我们平均只需要 2.5 比特来传输一个字符。

p	t	k	a	i	u
100	00	101	01	110	111

^① 波利尼西亚语与夏威夷的方言一样,都以其简短而著称。

前面的编码遵循了一个原则,即尽可能使用较少的比特位数传送频率高的字符。但是,在解码时并没有任何歧义产生,当编码以 0 开始时,这个编码长度为 2,当编码以 1 开始时,编码长度为 3。当然,按照信息论原理实际设计编码方案时,还需要做很多其他的工作,但这不是我们讨论的重点,有兴趣的读者可以参考相关书籍。

我们可以用熵的理论解释很多问题。有一类题目是通过提问来确认庄家手里的答案,并要求庄家对提出的问题只能做出肯定或者否定的回答,例如“这个字符是 t 还是 a ”,“它是一个常数吗?”按照这个理论,我们需要平均提出 2.5 个问题才能确认一个波利尼西亚语字符(假设所提出的问题是恰当的问题)。换言之,熵还可以作为衡量随机变量取值和相关概率的搜索空间大小的一种手段。

按照熵的定义,它具备以下 3 个属性:

1. $H(X) \geq 0$;
2. $H(X) = 0$, 当且仅当随机变量 X 的值是确定的,没有任何信息量可言;
3. 熵值随着信息长度的增加而增加。

例如,在抛非均匀硬币的实验中,我们传输实验结果所用的信息长度取决于硬币正面朝上的概率和抛硬币的次数。以抛一次硬币的实验为例,我们用图 2.5 形象地说明了不同概率下的熵值大小,同理对于多次实验的情况,因为每次实验是独立的,所以只需要把熵值和实验次数相乘。

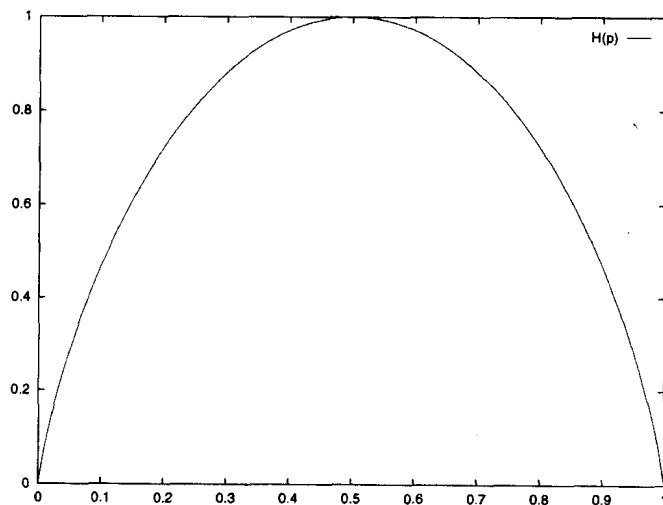


图 2.5 抛非均匀硬币事件的熵值。横轴表示硬币正面朝上的概率,竖轴表示抛一次硬币实验的熵值

2.2.2 联合熵和条件熵

如果 (X, Y) 是一对离散随机变量,其联合概率分布密度函数为 $p(x, y)$, (X, Y) 的联合熵 $H(X, Y)$ 定义为:

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y) \quad (2.29)$$

如果离散随机变量 (X, Y) 的联合概率分布密度函数为 $p(x, y)$, 已知随机变量 X 的情况

下随机变量 Y 的条件熵,实际上表示的是在已知 X 的情况下,传输 Y 额外所需的平均信息量,它定义为:

$$\begin{aligned} H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) \\ &= \sum_{x \in X} p(x) \left[- \sum_{y \in Y} p(y|x) \log p(y|x) \right] \\ &= - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(y|x) \end{aligned} \quad (2.30)$$

熵的连锁规则为:

$$\begin{aligned} H(X, Y) &= H(X) + H(Y|X) \\ H(X_1, \dots, X_n) &= H(X_1) + H(X_2|X_1) + \dots + H(X_n|X_1, \dots, X_{n-1}) \end{aligned} \quad (2.31)$$

其证明过程如下:

$$\begin{aligned} H(X, Y) &= -E_{p(x, y)}(\log p(x, y)) \\ &= -E_{p(x, y)}(\log(p(x) p(y|x))) \\ &= -E_{p(x, y)}(\log p(x) + \log p(y|x)) \\ &= -E_{p(x)}(\log p(x)) - E_{p(x, y)}(\log p(y|x)) \\ &= H(X) + H(Y|X) \end{aligned}$$

下面我们对例 8 中的波利尼西亚语言的例子略加修改。

例 9 科学的理论和实际应用之间总是存在差异,波利尼西亚语字符并不是随机变量,但是在例 8 中我们做了一个近似假设。进一步的研究发现,其实波利尼西亚语同样也具有音节结构,它的所有单词都由辅音和元音序列 CV 音节构成。在此基础上,我们可以构建更合理的模型,并用符号 C 表示音节中的辅音,用符号 V 表示音节中的元音,联合分布用 $P(C, V)$ 表示,边缘分布用 $P(C, \cdot)$ 和 $P(\cdot, V)$ 表示,分别定义如下:

	p	t	k	
a	$\frac{1}{16}$	$\frac{3}{8}$	$\frac{1}{16}$	$\frac{1}{2}$
i	$\frac{1}{16}$	$\frac{3}{16}$	0	$\frac{1}{4}$
u	0	$\frac{3}{16}$	$\frac{1}{16}$	$\frac{1}{4}$
	$\frac{1}{8}$	$\frac{3}{4}$	$\frac{1}{8}$	

(2.32)

其中,边缘概率的值是相对于一个音节的,因此它们的值为基于字符得出的概率值的两倍:

$$\begin{array}{cccccc} p & t & k & a & i & u \\ 1/16 & 3/8 & 1/16 & 1/4 & 1/8 & 1/8 \end{array} \quad (2.33)$$

当然,计算联合熵的方法很多,下面我们利用连锁规则计算联合熵的值^①:

① 在计算公式中,我们采用了一种简便表示离散分布的非正式方法,将它们拆解为概率值累加的形式,分别单独计算每个概率的熵。

$$\begin{aligned}
 H(C) &= 2 \times \frac{1}{8} \times 3 + \frac{3}{4} (2 - \log 3) \\
 &= \frac{9}{4} - \frac{3}{4} \log 3 \text{ 比特} \approx 1.061 \text{ 比特}
 \end{aligned}$$

$$\begin{aligned}
 H(V|C) &= \sum_{c=p,t,k} p(C=c) H(V|C=c) \\
 &= \frac{1}{8} H\left(\frac{1}{2}, \frac{1}{2}, 0\right) + \frac{3}{4} H\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\right) + \frac{1}{8} H\left(\frac{1}{2}, 0, \frac{1}{2}\right) \\
 &= 2 \times \frac{1}{8} \times 1 + \frac{3}{4} \left[\frac{1}{2} \times 1 + 2 \times \frac{1}{4} \times 2 \right] \\
 &= \frac{1}{4} + \frac{3}{4} \times \frac{3}{2} \\
 &= \frac{11}{8} \text{ 比特} = 1.375 \text{ 比特}
 \end{aligned}$$

$$\begin{aligned}
 H(C, V) &= H(C) + H(V|C) \\
 &= \frac{9}{4} - \frac{3}{4} \log 3 + \frac{11}{8} \\
 &= \frac{29}{8} - \frac{3}{4} \log 3 \approx 2.44 \text{ 比特}
 \end{aligned}$$

我们得到的最终熵值为 2.44 比特, 相对于前面的 2.5 比特要小一些, 是什么原因导致了熵的减少呢? 我们可以将其归结为模型的不确定性的减少。

因为信息量的多少和信息的长度有关, 所以下面正式定义词/字符熵的概念。假设信息长度为 n , 则词/字符熵, 又称熵率 (entropy rate)^①, 定义如下:

$$H_{\text{rate}} = \frac{1}{n} H(X_{1n}) = -\frac{1}{n} \sum_{x_{1n}} p(x_{1n}) \log p(x_{1n}) \quad (2.34)$$

语言可以看做一个随机序列 $L = (X_i)$, 比如我们的日常用语就可以看做这样一个序列, 报纸中的所有新闻报道文章也可以看做这样一个随机序列。我们可以定义这类语言的熵率计算公式, 即对语言样本的熵率取极值, 其中 n 为样本子集的长度。

$$H_{\text{rate}}(L) = \lim_{n \rightarrow \infty} \frac{1}{n} H(X_1, X_2, \dots, X_n) \quad (2.35)$$

2.2.3 互信息

根据熵的连锁规则, 我们有如下的计算公式:

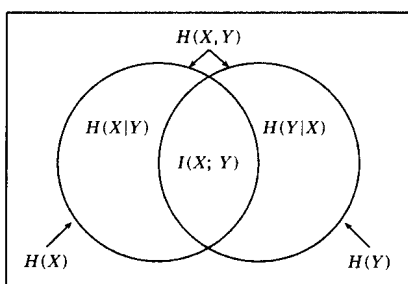
$$H(X, Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$$

所以有:

$$H(X) - H(X|Y) = H(Y) - H(Y|X)$$

这个差值称为随机变量 X 和 Y 之间的互信息 (mutual information), 用 $I(X; Y)$ 表示, 实际上它表示在已知 Y 的值后 X 的不确定性的减少量, 即随机变量 Y 揭示了多少关于 X 的信息量。图 2.6 用图示的方法直观地说明了互信息和熵之间的关系, 摘自 (Cover and Thomas 1991:20)。

① 本书中一般用带两个下标的变量表示子序列, 例如这里用 X_{ij} 表示随机变量序列 $(X_i, \dots, X_j)$, 即 $x_{ij} = (x_i, \dots, x_j)$ 。虽然人们一般很少使用这种记号, 但是它可以很方便地表示子序列。

图 2.6 互信息 I 和熵 H 之间的关系

互信息是一种计算两个随机变量之间共有信息的度量,满足非负性和对称性两个特点。虽然直观地看互信息可以衡量变量之间的依赖程度,但实际上它更应该作为验证变量是否互不相关的手段,因为它的取值具备以下两个特点:(1) 当两个随机变量无关时,互信息为零;(2) 当变量之间存在依赖关系时,它们的互信息不仅和依赖程度相关,而且和变量的熵也相关。利用简单的数学推导可以得到以下的计算公式^①:

$$\begin{aligned}
 I(X; Y) &= H(X) - H(X|Y) \\
 &= H(X) + H(Y) - H(X, Y) \\
 &= \sum_x p(x) \log \frac{1}{p(x)} + \sum_y p(y) \log \frac{1}{p(y)} - \sum_{x,y} p(x, y) \log p(x, y) \\
 &= \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}
 \end{aligned} \quad (2.36)$$

显然, $H(X) = H(X) - H(X|X) = I(X; X)$, 因为 $H(X|X) = 0$, 这解释了熵又称为自互信息的原因, 同时也验证了上面的论点, 即完全依赖关系的两个变量之间的互信息不是常数, 而是一个依赖于熵的具体取值。

同理, 我们推导出条件熵的计算公式及其相应的连锁规则:

$$I(X; Y|Z) = I((X; Y)|Z) = H(X|Z) - H(X|Y, Z) \quad (2.37)$$

$$\begin{aligned}
 I(X_{1:n}; Y) &= I(X_1; Y) + \cdots + I(X_n; Y|X_1, \dots, X_{n-1}) \\
 &= \sum_{i=1}^n I(X_i; Y|X_1, \dots, X_{i-1})
 \end{aligned} \quad (2.38)$$

本节中定义了两个随机变量之间的互信息。有时候, 我们还应用到点互信息 (pointwise mutual information) 的概念, 即计算分布中两个特定样本点之间的互信息。

$$I(x, y) = \log \frac{p(x, y)}{p(x)p(y)}$$

它通常用来计算两个元素之间的关联程度。有一些问题将在 5.4 节中讨论。

▼ 互信息在统计自然语言处理中应用得很广泛, 例如可以用于词的聚类 (见 14.1.3 节) 和语义消歧 (见 7.2.2 节)。

① 在有关互信息的公式中, 两个变量通常用分号隔开。

2.2.4 噪声信道模型

香农借助于信息论的概念,概括描述了信道通信的目标,即在信道存在噪声的情况下,优化通信信息的吞吐量和准确率。我们假设在统计上信道的输出依赖于输入。一般来说,在压缩率和传输的准确率之间存在着矛盾,即我们利用去除冗余信息的方法实现压缩,同时又通过加入可控的额外信息保证传输的准确率。所以,最终的编码遵循这样一种编码原则,既要占用最少的信道空间,又必须包含足够的冗余信息以检测和纠正错误。在接受方,传输的信息被解码,还原为初始信息,这个过程可以用图 2.7 表示。

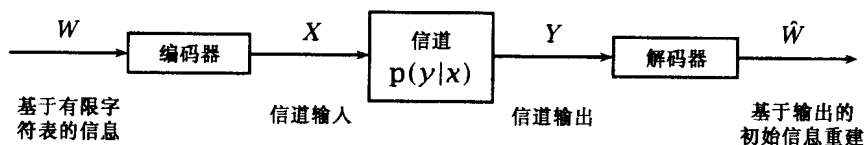


图 2.7 噪声信道模型

在信息论中,刻画信道特性的最重要的概念是信道容量(capacity),信道容量表明信道在保证任意低的传输错误率的情况下传输信息的速率。对于一个无记忆性信道而言,香农第二定理表明信道容量可以用互信息表示:

$$C = \max_{p(X)} I(X; Y) \quad (2.39)$$

按照这个定义,我们如果能够设计一个编码方案,它的概率分布 $p(X)$ 可使上式中的输入与输出之间的互信息取最大值,那么传送速率就可以达到信道容量。

举例说明,如图 2.8 中的二元对称信道,输入只限定为“0”或者“1”,信道中的噪声可能影响正确的输出,使“1”变“0”或者“0”变“1”。设这种事件发生的概率为 p ,我们发现:

$$\begin{aligned} I(X; Y) &= H(Y) - H(Y|X) \\ &= H(Y) - H(p, 1-p) \end{aligned}$$

如上所述,互信息的最大化可以通过最大化编码的熵来实现,因此我们可以假设输入为均匀分布,这样输出同样也为均匀分布, $H(Y)$ 的值等于 1,我们得到:

$$\max_{p(X)} I(X; Y) = 1 - H(p, 1-p)$$

同时,熵值满足非负性,所以 $C \leq 1$ 。在上式中当且仅当熵值为零时信道容量为 1,相当于概率取 0 或者 1,即信息的输入和输出完全相同或者输入和输出完全相反。也就是说,当 $p=0$ 时, $0 \rightarrow 0, 1 \rightarrow 1$; 当 $p=1$ 时, $0 \rightarrow 1, 1 \rightarrow 0$ 。由于噪声的影响,当信道输入为“0”,“1”,输出分别为“0”,“1”的概率为 $1/2$ 时,信道容量 C 为 0。这样一个信道对于传输数据没有任何用处,因为它的输出具有最大的不确定性。

香农的信息论之所以至关重要,从而作为信息论的早期“胜利”成果,关键在于两点。第一点,它定义了信息容量的概念,换言之,它给出了一个传输速率 $I(X; Y)$ 的最小上界(设输入分布为 $p(X)$)。第二点,它指出在实际应用中我们可以设计出一个适当的编码,使信道的传输率接近或者等于最优。信息容量概念的提出消除了以前人们在设计通信系统时采用的一些猜测

性的方法,我们可以根据香农的理论精确地评估系统设计或者编码方案的好坏,判断它们是否具有最佳或者近似最佳的性能。

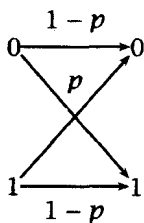


图 2.8 一个二元对称信道模型。输入为“0”或者“1”，对应的输出为“1”和“0”，输入和输出取相反值的可能性概率为 p

噪声信道理论在统计自然语言处理中很重要,甚至可以说在一定程度上它标志了 20 世纪 70 年代量化自然语言处理方法的兴起。例如,在 20 世纪 50 年代到 20 世纪 60 年代,继首个大型的基于量化的自然语言处理项目之后,IBM T. J. Watson 研究中心首先使用了噪声信道模型来构建语音识别和机器翻译系统。

通过将噪声信道模型用于语言学中,我们希望解码模型的输出近似于初始输入,而且并不涉及到模型编码阶段。我们采用的模型如图 2.9 所示。自然处理中的很多问题都可以被映射为此类问题,即给定输出结果后,寻找最有可能的输入。我们可以通过贝叶斯理论将这个过程定义如下,其中输出概率是一个常数:

$$\hat{i} = \arg \max_i p(i|o) = \arg \max_i \frac{p(i) p(o|i)}{p(o)} = \arg \max_i p(i) p(o|i) \quad (2.40)$$

公式中出现了两个概率分布, $p(i)$ 是语言模型,表示输入语言词序列的分布, $p(o|i)$ 表示信道概率。

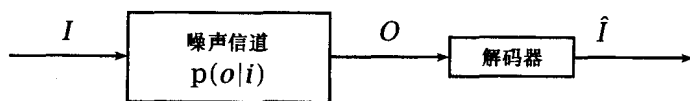


图 2.9 语言学中的噪声信道模型

举例说明,假设我们希望将文本从英文翻译成法文,机器翻译的噪声信道模型假设输出的真实文本为法文,通过一个噪声信道最终传输给我们的却像是英文,这可以看做受到了噪声的影响,即这种变化是由噪声引起的。所以,我们在输出端得到的是英文单词 *cow*,但事实上它应该是法文单词 *vache*,依照这个定义,机器翻译的目的就是将英文解码为法文^①。

机器翻译的噪声信道模型的正确与否一直是自然语言处理研究者们争论的焦点,但不可否认,它满足数学定义的严格性,并且在此基础上有很多重要的研究成果,在第 13 章中将重点讨论这个问题。当然,统计自然语言处理中还有很多其他问题都可以被看做解码的过程,表 2.2 给出了一个列表。

① 以法语为母语的人也许认为英语实际上是一种被曲解了的法语,使语言产生毫无意义的歧义。

表 2.2 等同于解码过程的统计自然语言处理问题

应用	输入	输出	$p(i)$	$p(o i)$
机器翻译	L_1 词序列	L_2 词序列	$p(L_1)$ 基于语言模型	翻译模型
光学字符识别(OCR)	实际文本	输出的含有错误的文本	语言文本概率	OCR 错误模型
词性标注(POS)	英语文本	POS 标记序列	POS 序列概率	$p(w i)$
语音识别	语音信号	词序列	词序列概率	声学模型

2.2.5 相对熵和 Kullback-Leibler 距离

给定两个概率密度函数 $p(x)$ 和 $q(x)$, 它们的相对熵(relative entropy)又称为 Kullback-Leibler(KL)距离, 由下式给出:

$$D(p \| q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} \quad (2.41)$$

在定义中我们约定 $0 \log(0/q) = 0$, $p \log(p/0) = \infty$ 。

式(2.41)可以表示成期望值形式:

$$D(p \| q) = E_p \left(\log \frac{p(X)}{q(X)} \right) \quad (2.42)$$

因此, 从信息论角度看, 如果一个随机变量 X 的分布为 p , 但我们却以概率分布 q 来给它编码, 那么相对熵就是这样的编码比按照概率分布 p 的编码多使用的比特位。

相对熵满足两个属性, (1) $D(p \| q) \geq 0$, 等式当且仅当 $p(x) = q(x)$ 时成立; (2) $D(p \| q) \neq D(q \| p)$, 即非对称性, 并且不满足三角不等式^①。相对熵表示了两个随机分布之间的差异程度, 当两个随机分布的差别增大时, 它们的相对熵也增大, 所以它通常被用来衡量随机分布之间的近似程度。

互信息也可以从相对熵角度进行解释, 从这个角度上看实际上表示联合分布和假设独立分布之间的差异程度, 因为互信息可以写成如下形式:

$$I(X; Y) = D(p(x, y) \| p(x)p(y)) \quad (2.43)$$

仿照前面的定义, 我们同样可以推导出条件相对熵(conditional relative entropy)及其连锁规则的计算公式(参见(Cover and Thomas 1991:23)):

$$D(p(y|x) \| q(y|x)) = \sum_x p(x) \sum_y p(y|x) \log \frac{p(y|x)}{q(y|x)} \quad (2.44)$$

$$D(p(x, y) \| q(x, y)) = D(p(x) \| q(x)) + D(p(y|x) \| q(y|x)) \quad (2.45)$$

▼ KL 距离用于选择倾向的量测的内容见 8.4 节。

2.2.6 交叉熵

到目前为止, 我们已经讨论了熵的概念, 说明其在信息传送中的作用, 熵的概念同样也可

① 任意给定三个点 (x, y, z) , 都满足:

$$d(x, y) \leq d(x, z) + d(z, y)$$

以引入到语言理解问题上。我们知道熵可以用于衡量事件不确定性的尺寸,事件所透露出来的信息越多,其熵值会越小。

这里仍以前面例 8 和例 9 中的波利尼西亚语为例进行说明,它的字符表中有 6 个字符。最简单的编码方式是用 3 位二进制数表示各个字符,等同于假设了一个满足均匀概率分布的语言模型,但是实际上字符的出现次数不是平均的,所以以频度信息表示字符相当于建立了一个零阶的语言模型。此时,它的字符熵(perletter)大致为 2.5。此后,我们注意到波利尼西亚语也有音节结构,并在此基础上构建了一个结合音节信息的语言模型,它的字符熵大致为 1.22。相比较而言,这个模型更好地利用了语言的内部结构特征,因此它的熵比较小,换言之,熵的大小可以作为评估语言模型好坏的一个标准。

另一方面,我们可以把熵看做对某个事件发生所持的“惊奇”程度,“惊奇”程度越大说明该事件的发生越出乎意料。举例说明,假设我们的目的是预测当前波利尼西亚语文本中下一个词出现的概率,也就是计算 $P(w|h)$ 的大小, h 代表当前已知的文本, w 代表下一个词。我们观察到下一个词出现时的“惊奇”程度可用条件概率的大小来描述,假设模型用 m 表示,定义点态熵(pointwise entropy) $H(w|h) = -\log_2 m(w|h)$ 。已知 h , 如果确定词 w 出现的概率等于 1, 那么 $-\log_2 1 = 0$, 也就是说,它没有给我们带来任何的“惊奇”。而如果模型认为词 w 不会在 h 后出现,即 $m(w|h) = 0$, 那么 $-\log_2 0 = \infty$, 换句话说,“惊奇”度趋向于无限大,这不是我们所期望的。这些都是极端的情况,一般的熵值位于两者之间,如果计算每个词相对于前面的文本出现的概率,并将这些值累加,最终结果为“惊奇”程度的总和,用熵来表示:

$$\begin{aligned} H_{\text{total}} &= -\sum_{j=1}^n \log_2 m(w_j | w_1, w_2, \dots, w_{j-1}) \\ &= -\log_2 m(w_1, w_2, \dots, w_n) \end{aligned}$$

公式的推导过程应用了连锁规则,如果我们利用文本长度对上式进行归一化处理,其结果表示平均每个词给我们带来的“惊奇”程度,并且这个值不再和文本的大小有关。

上述讨论的目的是希望给读者一个直观的印象,是一种非正式的定义。借助相对熵的概念,我们可以给出正式的定义。假设已知语言样本先验分布,例如在某种语言的言语内容中,每个词都可以映射到数字,我们就可以用随机变量 $X \sim p(x)$ 描述它们。显然,一些变量发生的概率要大于另外一些变量发生的概率,例如在日常生活中我们听到 *thank you* 的频率肯定比 *On you* 要大。

遗憾的是, $p(\cdot)$ 的确切分布是未知的,但是通过已知实例,例如一定规模的语音语料库,我们可以大致估计出 p 的分布。在构建该模型时极小化 $D(p \parallel m)$ 的值,即让估计的模型和真实模型 m 之间的概率分布的差最小,但是仍不能直接计算这个相对熵的大小,因为不知道 p 的大小。幸运的是,我们可以得到一个相对的量,即交叉熵。

设 $X \sim p(x)$, $q(x)$ 为我们估计的近似 $p(x)$ 的一个概率分布,则 $p(x)$ 和 $q(x)$ 的交叉熵表示为:

$$\begin{aligned} H(X, q) &= H(X) + D(p \parallel q) \\ &= -\sum_x p(x) \log q(x) \end{aligned} \quad (2.46)$$

$$= E_p\left(\log \frac{1}{q(x)}\right) \quad (2.47)$$

具体的证明留给读者,参见习题 2.13。

在 2.2.2 节中定义了语言的熵的概念,同样我们可以定义一个语言的交叉熵 $L = (X_i) \sim p(x)$, 设模型用 m 表示,则有:

$$H(L, m) = - \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{x_{1n}} p(x_{1n}) \log m(x_{1n}) \quad (2.48)$$

从上式直接计算交叉熵的大小仍然存在一定的难度,因为到目前为止,我们对 p 的确切分布仍一无所知,但是我们可以假设该语言是一种定义“良好”的语言,那么它的交叉熵就可以写为:

$$H(L, m) = - \lim_{n \rightarrow \infty} \frac{1}{n} \log m(x_{1n}) \quad (2.49)$$

按照上式的定义,计算交叉熵只需要一个大规模的样本数据,而且只要保证 n 足够大,就可以得到近似的结果:

$$H(L, m) \approx - \frac{1}{n} \log m(x_{1n}) \quad (2.50)$$

式(2.50)给出了交叉熵的计算公式,衡量了所谓的平均“惊奇”度。我们的目标是使这个数值最小。因为 $H(X)$ 的值一般是固定不变的,因此交叉熵的最小化等同于相对熵的最小化,即估计出概率分布和真实数据分布之间的差值。这里,另一个要求是用来测试模型的文本必须是一个独立的测试数据集。这个数据集不能是用来估计模型参数的训练语料库的一部分。交叉熵和模型赋予测试数据集中的词的平均概率反向相关。模型的交叉熵越低,一般就意味着它在应用中的性能越好;如果只是为了改进概率估计的量,而不是概率估计之间的相关次序,就可以不必计算交叉熵。参阅 6.2.3 节可以了解有关如何计算交叉熵的更多细节。

但是读者肯定会提出一个疑问:式(2.48)是如何转变到式(2.49)的呢?下面我们简单介绍一下其中的推导过程。式(2.48)可以写成如下期望值的形式:

$$H(L, m) = \lim_{n \rightarrow \infty} \frac{1}{n} E \left(\log \frac{1}{m(X_{1n})} \right) \quad (2.51)$$

我们知道,期望值是对所有可能序列的加权平均,式(2.51)是一个求极限公式,其基本思路是当语言样本很大时不必计算加权平均,按照当前这个“典型”样本计算出的熵率可作为近似结果。

按照 Shannon-McMillan-Breiman 定理(又称渐进均分性质, Asymptotic Equipartition Property), 如果假设 $L = (X_i)$ 是一个平稳遍历过程(stationary ergodic process), 那么上面的结论存在严格的数学证明。

定理: 设 H_{rate} 是一个有限值的平稳遍历过程 (X_n) 的熵率, 那么

$$- \frac{1}{n} \log p(X_1, \dots, X_n) \rightarrow H$$

当 n 无限大时,这两个值之间的逼近概率为 1。

读者如果想了解详细的证明过程,可以参阅(Cover and Thomas, 1991: ch.3, 15)。直观地看,遍历过程是一个各态历经的过程。为了说明这一点,我们举一个非遍历过程的例子。假设该过程以两个状态中的一个状态开始,这两个状态中的一个永远生成 0,另外一个永远生成 1,那么显然无论经过多长时间,我们都不可能获知这个过程的典型行为特征。不可能获知过程

重新开始时将会发生什么。

平稳过程具备时间无关性的特点,显然语言并不满足这个要求,因为语言中不断有新的用语加入,并且一些过时的表达方式会逐渐被淘汰。但是,从某一段特定的时间来看,假设语言特性不变,还是基本上近似正确的。这种假设方法也是我们经常采用的方法。

2.2.7 英语的熵

事实上英语并不严格满足平稳遍历过程的要求,但我们仍然依照随机过程对其建模,例如最常用的 n 元语法模型或者马尔可夫链,相关内容将在第 6 章和第 9 章中讨论,其中都事先假设了模型具有有限的记忆性。当我们假设下一个出现的词仅和前面 k 个词相关时,则相当于定义了一个 k 阶马尔可夫链。

$$P(X_n = x_n | X_{n-1} = x_{n-1}, \dots, X_1 = x_1) = P(X_n = x_n | X_{n-1} = x_{n-1}, \dots, X_{n-k} = x_{n-k})$$

如果在字符层面上讨论这个问题,例如已知文本中的前 k 个字符,我们会试着猜出下一个出现的字符。所以当一些印刷文本出于某些原因,其中缺失了一些字符或者出现了字迹模糊的情况时,由于英语的冗余性,重新填充这些字符对我们来说其实并不是件很困难的事情。

通过统计英语中单个字符的频率和两字符串(两个字符的序列)的频率,我们可以计算英语熵的上界值^①,因为由式(2.46)可知 $D(p \parallel m) \geq 0$, $H(X, m) \geq H(X)$,香农为了估算英语的熵,假设英语是由 27 个符号组成的语言(26 个字母和一个空格,忽略大小写和标点符号),得到了以下结果:

模型	交叉熵(单位为比特)	
零阶	4.76	(一致模型,因此为 $\log 27$)
一阶	4.03	
二阶	3.32	
香农实验	1.3(1.34)	(Cover and Thomas 1991:140)

(2.52)

实验表明,模型的阶数和模型的性能成正比,阶数越高,交叉熵越小。香农为了获取一个更好的英语熵上界,使用了人工实验加以推导,即寻找人类在猜文本中下一个字是什么时能猜得多好,结果实验得出的英语熵大概为 1.34 左右。这是英语熵较低的下界。

当然,英语的实际熵可能还要小,因为人们在日常的谈话中其实存在很多固定的模式,而且其中很多都是我们没有意识到的。到目前为止,所有我们可以构造出来的统计语言模型都比不上人类自身的语言能力,所以现阶段我们的目标是建立这样一种模型,它能够像英语说话者那样知道什么是正常的、好的英语,什么是蹩脚的、错误的英语。

▼ 第 6 章将会详细讨论语言模型的问题。

2.2.8 混乱度

在语音识别领域中,人们通常用混乱度(Perplexity)而不是交叉熵来描述一个模型的好坏,

^① 严格地说,我们要求这些频率值能反映出英语的特性,在字符层面上进行统计从某种意义上说问题还不小,但是如果通过统计词来计算英语的熵就会出现一些问题,第 4 章将会讨论这个问题。

虽然两者在本质上是相同的:

$$\text{perplexity}(x_{1n}, m) = 2^{H(x_{1n}, m)} \quad (2.53)$$

$$= m(x_{1n})^{-\frac{1}{n}} \quad (2.54)$$

我们猜想他们之所以应用混乱度而不是交叉熵是出于争取赞助资金的需要, 毕竟“我们将混乱度从 950 降到 540”要比“我们将交叉熵从 9.9 比特降到 9.1 比特”更具震撼性。虽然如此, 混乱度的概念更能给我们直观的感觉, 混乱度为 k 意味着我们在每一步中都面临着 k 个等概率的选择。

2.3 深入阅读

Aho et al. (1986: ch. 4) 从计算机科学的角, 而 Allen (1995: ch. 3) 则从计算语言学的角介绍了句法分析。除了向量空间和矩阵之外, 我们使用的大部分数学都可在 (Cormen et al. 1990) 中的第一部分找到。有关向量空间和矩阵的内容可参考线性代数方面的书籍, 如 (Strang 1988)。

基本概率论方面的参考书很多, 比如 (Moore and McCabe 1989), (Freedman et al. 1998), (Siegel and Castellan 1988) 和 (DeGroot 1975) 都是很好的读物 (近似按从易到难排序)。绝大多数统计学书籍的主要内容都是假设检验, 但遗憾的是, 书中的例子一般都来源于生物学或者心理学, 而和统计自然语言处理毫不相关。对于自然语言处理的研究者来说, (Krenn and Samuelsson, 1997) 是一个更好的选择, 它详细阐述了和自然语言处理问题相关的统计概率知识, 此外阅读一些机器学习方面的数据也有益于理解这些量化方法, 我们推荐阅读 (Mitchell 1997)。

本章只是简单地介绍了信息论的基本定义和概念, 读者如果希望进一步了解这方面的知识, 可以阅读 (Cover and Thomas 1991)。

另外, Brown et al. (1992b) 曾经在一个大规模的英文语料上估计英语的熵, 其结果大概为每个字符 1.75 比特。

2.4 习题

2.1 [★] 下面列出了几个在本书中经常使用的概率论公式, 试利用集合论和概率论的公理证明之。

a. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ [加法律]

b. $P(\emptyset) = 0$

c. $P(\bar{A}) = 1 - P(A)$

d. $A \subseteq B \Rightarrow P(A) \leq P(B)$

e. $P(B - A) = P(B) - P(A \cap B)$

2.2 [★] 设有样本空间

$$\Omega = \{\text{is-noun, has-plural-s, is-adjective, is-verb}\} \quad (2.23)$$

函数 $f: 2^\Omega \rightarrow [0, 1]$ 有如下取值:

x	$f(x)$
{is-noun}	0.45
{has-plural-s}	0.2
{is-adjective}	0.25
{is-verb}	0.3

函数 f 可否扩展到所有的 2^0 空间上, 并有一个合适的概率分布? 如果不能, 如何用概率来模写这些数据?

- 2.3 [★] 设事件 A 定义为“句点 '.' 出现在 3 字词后, 并且这个句点表示一种省略形式而不是表示句子结束”。

假设有:

$$P(\text{is-abbreviation} \mid \text{three-letter-word}) = 0.8 \quad (2.24)$$

$$P(\text{three-letter-word}) = 0.0003 \quad (2.25)$$

计算事件 A 的发生概率。

- 2.4 [★] 假设离散随机变量 X 和 Y 的分布定义为:

x	0	0	1	1
y	0	1	0	1
$p(X=x, Y=y)$	0.32	0.08	0.48	0.12

试问它们的分布是否独立?

- 2.5 [★] 在例 5 中, 我们计算了两个骰子点数和的期望值, 但采用的是单独计算每个骰子期望值的方法。证明直接计算两个骰子点数的期望值和本章例 5 的结果是相同的。

- 2.6 [★★] 收集自己在过去两年内所修课程的成绩, 并把它们转化为合适的数字, 试问用什么分布来描写它们?

- 2.7 [★★] 在自然语言中寻找一种“语言现象”, 使之可以用二项分布描述, 并且给出参数 p 的最佳估计?

- 2.8 [★★] 当 $i=8, j=2$ 时, 式(2.15)的最大值为 0.8, 式(2.17)的最大值为 0.75, 假设我们用下式来表示模型的先验知识:

$$P(\mu_m) = 30m^2(1-m)^2$$

那么在投硬币实验中, 当我们观察到 10 次实验中 8 次正面朝上, 2 次正面朝下时, 试估计 MAP 概率值(最大后验概率)。假设硬币均匀, 模型为 μ_m 。

- 2.9 [★] 选取一段短文本, 计算文本中字母的相对频率, 假设这些都是实际的概率分布, 试给出该分布的熵。

- 2.10 [★] 另外选取一段短文本, 用相同的方法计算它的概率分布, 试给出该分布和习题 2.9 中概率分布之间的 KL 距离。可能需要对该习题中的分布进行一些平滑, 用一个小概率值 ϵ 代替那些零概率事件。

- 2.11 [★] 将语义消歧问题映射到噪声信道模型中。可以参照表 2.2 中的例子, 例如确定 *industrial plant* 和 *living plant* 中单词 *plant* 的确切含义。第 7 章要讨论语义消歧的问题。

- 2.12 [★]试举一个例子说明 KL 距离的非对称性,例如对于两个分布函数 p 和 q , $D(p \parallel q) \neq D(q \parallel p)$ 。
- 2.13 [★] 证明式(2.46)的正确性。
- 2.14 [★]式(2.49)给出了交叉熵的计算方法,但它要求我们处理的过程是一个平稳遍历过程,试举出自然语言的一些特性,说明英语只是近似满足这两个属性。
- 2.15 [★★]重复香农所做的实验,编写一个英文文本操作程序,每次显示文本中的一个字符,在一个未知文本上运行该程序,观察是否能够验证香农对英语的熵的估计值。
- 2.16 [★★] 重复习题 2.15 的工作,但不同的是我们需要选取两篇文本,一篇是容易阅读的文档,比如新闻报道,另外一篇则比较晦涩,比如科学论文,下面提出两个问题:
1. 它们各自的熵的估计值是否相同?
 2. 如果估计值不同,试说明其原因。

第3章 语言学基础

本章介绍了一些基本的语言学概念,这些概念对于了解本书后面部分的讨论都非常重要。这里不仅回顾了学校所学的部分知识,同时也深入讨论了一些语法现象,比如附着歧义和短语结构这些在 NLP 中很重要的概念。除了语法(句子结构)外,我们还将涉及到词法(词语形成)和语义。最后一节将会给出语言学其他一些领域的概述和深入阅读材料。

3.1 词性和词法

语言学家们将词按照相似的语法结构行为和典型的语义类型聚成不同的类(集合)。这些词的类别被称为句法或语法类,但是更常用的仍然是传统的名称,即词性(parts of speech,简称 POS)。通常说来有三类重要的词性:名词、动词和形容词。名词特指人、动物、概念或事物。动词的最初用法是用来表达句子中的动作。形容词描述了名词的属性。判断词是否属于同一类的最基本方法是替换测试。形容词可以是例句(3.1)大括号中的词的任何一个:

$$\text{The } \left\{ \begin{array}{l} \text{sad} \\ \text{intelligent} \\ \text{green} \\ \text{fat} \\ \dots \end{array} \right\} \text{ one is in the corner.} \quad (3.1)$$

在例句(3.2)中,名词 *children* 指一群人(年纪小),名词 *candy* 是一种特殊的食品:

$$\text{Children eat sweet candy.} \quad (3.2)$$

动词 *eat* 描述了 *children* 对于 *candy* 的动作。形容词 *sweet* 告诉了我们 *candy* 的一种属性,即它是 *sweet* 的。许多词都存在兼类情况,例如 *candy* 也能作为动词(*Too much boiling will candy the molasses*),而且,我们知道,至少在英国英语中,*sweet* 可以被用做名词,意思大致和 *candy* 一样。词的类别通常分成两类。开放类别又称词汇类别,指的是名词、动词以及形容词这些有大量成员的词,同时还不断地有新词加入。封闭类别又称功能类别,指的是介词、限定词(包含类似 *of, on, the, a* 的词)等,它们只有少量的成员,而且通常也有比较清楚的语法用途。通常,在线词典会列出一个词的不同词性。

传统的词性标注系统区分了 8 种类别,但是语料库语言学家们则希望使用更加细致地对词语类别进行分类的方法。目前已经很好地建立了命名这些类别的缩写集,通常称之为 POS 标记。在本章中介绍语法结构类别时,会给出 Brown 语料库中对于那些最重要的词类的缩写。例如,在 Brown 语料库中,形容词标记用代码 JJ 表示。因为 Brown 语料库在语料库语言学中的先锋角色,它的标记系统已经得到了广泛的认同。

▼ 对几种著名的词性标记集的描述和比较见本书的 4.3.2 节。

词语类别与构词过程是系统相关的,例如从名词单数形式(*dog*)变成复数形式(*dogs*)的过

程。词法在 NLP 中是非常重要的,因为语言是可以生成的,在任何给定的文本中都会遇到以前从未见过或没有包含在现有词典中的词或词语形式。这些新词在词法上都与已知的词是相关的。因此,如果我们懂得了构词过程,就能够推导出新词的许多语法和语义属性。

在英语中掌握构词法是很重要的,但是对于芬兰语,掌握构词法是绝对需要的。在英语中,一个规则动词只有 4 种不同的形式,不规则动词则最多有 8 种形式。我们可以不采用构词法而通过列出所有词的形式来完成相当多的一部分。与之形成对比的是,芬兰语中的一个动词可以有超过 10 000 种形式!所以在处理类似芬兰语的语言时,把所有的动词形式枚举成一个大的列表是冗长并且不可行的。

构词过程的主要类型有:变形、派生或复合。变形是对词根形式进行系统的修改,通过加前缀或者后缀来指明语法结构的不同,比如单数和复数。变形并不显著地改变词语的类别和语义,但是修改了一些特征,比如时态、数目或者复数形式。我们可以把一个词的所有变形组成一个单独的列表。

相对而言,派生变化就不是很系统化的过程,我们无法用一种通用的方法来概括它。它通常导致了语法类别的根本改变,而且涉及到了含义的变化。例如副词 *widely* 是从形容词 *wide* 变来的(通过加后缀 *-ly*)。 *widely* 在短语 *it is widely believed* 中的意思是“在一个大的很分散的人群中”,与 *wide* 的核心意思“延伸到一个很大的区域”已经有了变化。副词变形相对于复数变形来说更加不系统。一些形容词,比如 *old* 或 *difficult* 就没有副词形式, *oldly* 或 *difficultly* 并不是英语中的词。这里有一些派生的例子:后缀 *-en* 将形容词变为动词(*weak-en*, *soft-en*),后缀 *-able* 将动词变为形容词(*understand-able*, *accept-able*), *-er* 则将动词变为名词(*teach-er*, *lead-er*)。

复合词是指将两个或者多个词合成一个新词。英语中存在很多名词-名词复合词,或者是由另外两个名词组合而成的名词,例如 *tea kettle*, *disk drive* 或者 *college degree*。但是它们通常写成单独的词,而读成一个词,表示一个独立的语义概念,这些词在词典中通常都会列出。另外,还有一些其他同性的(例如形容词、动词和介词)的复合词,比如 *downmarket*, (*to*) *overtake* 以及 *mad cow disease*。

下面我们将介绍英语中的主要词性。

3.1.1 名词和代词

名词特指世界上的实体,例如人、动物和事物,如下所示:

$$\text{dog, tree, person, hat, speech, idea, philosophy} \quad (3.3)$$

相对于其他语言而言,英语构词法比较简单,变化很少,名词只有一种变形,即复数形式。通常通过加后缀 *-s* 来实现。下面就有一些名词的单复数形式:

$$\begin{array}{lll} \text{dog : dogs} & \text{tree : trees} & \text{person : persons} \\ \text{hat : hats} & \text{speech : speeches} & \text{woman : women} \\ \text{idea : ideas} & \text{philosophy : philosophies} & \text{child : children} \end{array} \quad (3.4)$$

复数后缀有三种发音: *hats* 中读 /s/, *boys* 中读 /z/, *speeches* 中读 /əz/, 最后一个例子表示了书写系统中要插入 *e* 的情况。其他一些形式,比如 *women* 并不遵从这种规则,我们称之为不规则形式。

数(单数和复数)是名词的一种常见的语法差异。另外还有两种跨语言的常用的名词变形,即性别和格,都已经在表 3.1 中列出了。

表 3.1 常见的名词变形

变形类型	实例
数	<i>singular, plural</i>
性别	<i>feminine, masculine, neuter</i>
格	<i>nominative, genitive, dative, accusative</i>

英语中不存在性别变形形式,但是对于第三人称单数代词有不同的性别形式:he(男性),she(女性),it(中性)。一个由拉丁语而来的名词性别变形的例子是表示女性的名词后缀为-a,男性为-us。例如,*filius*(儿子,男孩),*filia*(女儿,女孩)。上面提到的英语中第三人称单数变形和拉丁语中的这两个单词类似(女性→阴性,男性→阳性,其他→中性)。在一些语言中,语法性别变形和所指人的性别密切相关,但是其他一些语言中性别是很随意的语法类别。语言学家们常举的一个例子是,德语中表示女孩的单词 *Mädchen* 是中性的。

在一些语言中,当名词在句子中具有不同的功能(主语,宾语等)时,该名词表现为不同的形式,这种形式称为格(case)。例如,拉丁语中儿子的主格是 *filius*,宾格是 *filium*。许多语言具有丰富的变格形式,譬如位置格、工具格,等等。英语没有真正的格变换。其中惟一系统指明的格关系是所有格。所有格描述了所有者。例如,短语 *the woman's house* 指明 *woman* 拥有 *house*。属格经常写为's,但是当单词以s结尾时写为',其中包括绝大多数名词复数,譬如 *the students' grievances*。虽然最初's被看成是一种格变换,但它实际上是一种接语(clitic),或者是熟知的短语词缀,因为它不仅能附着在名词上,还可以出现在名词的修饰语之后,譬如 *the person you met's house was broken into*。

代词是一个独立的小类,类似于变量,指代上下文中比较明显的人或物。例如,在例句(3.5)中代词 *she* 指代的是上下文中最明显的人物 *Mary*(女性)。

After *Mary* arrived in the village, *she* looked for a bed-and-breakfast. (3.5)

和代词所指代的对象区分单、复数一样,代词也区分人称(第一人称是发言者,第二人称是听众,第三人称是其他谈论对象)。这是英语中惟一一种在作为句子主语和宾语的时候形式不同的单词。这两种形式分别称为主格(主语格)或宾格(宾语格)人称代词。代词也具有特殊的形式,所有格代词中的代词指代的对象是一个所有者,例如 *my car*,我们可以认为它是所有格形式。有点奇怪的是,英语代词具有另一种所有格形式,通常称为第二所有格人称代词,用于介词 *of* 的宾语描述的是所有者时,如 *a friend of mine*。最后,还有反身代词,它们与普通的(人称)代词使用方法基本相同,所不同的是其指代同一句中的距离最近的对象,经常是句子的主语。例如,例句(3.6a)中的 *herself* 指代的一定是 *Mary*,而例句(3.6b)中的 *her* 不能指代 *Mary*,也就是说, *Mary* 在镜子中看见了另外一个女人。

- a. *Mary* saw *herself* in the mirror.
b. *Mary* saw *her* in the mirror. (3.6)

反身代词(以及其他特定的表达方式,如 *each other*)常用于指代语,指代文中某个和它距离非常接近的对象。人称代词则用来指代前文中稍远距离出现的人或物。代词的所有形式以及它们的 Brown 标记总结在表 3.2 中。

表 3.2 英语中的代词形式。除了反身代词以外,第二人称形式并没有区分数,但是第三人称单数形式区分了性别

	主格	宾格	所属格	第二所属格	反身的
标记	PPS(3SG) PPSS(2SG, 2SG, PL)	PPO	PP \$	PP \$ \$	PPL (PPLS for PL)
1SG	<i>I</i>	<i>me</i>	<i>my</i>	<i>mine</i>	<i>myself</i>
2SG	<i>you</i>	<i>you</i>	<i>your</i>	<i>yours</i>	<i>yourself</i>
3SG MASC	<i>he</i>	<i>him</i>	<i>his</i>	<i>his</i>	<i>himself</i>
3SG FEM	<i>she</i>	<i>her</i>	<i>her</i>	<i>hers</i>	<i>herself</i>
3SG NEUT	<i>it</i>	<i>it</i>	<i>its</i>	<i>its</i>	<i>itself</i>
1PL	<i>we</i>	<i>us</i>	<i>our</i>	<i>ours</i>	<i>ourselves</i>
2PL	<i>you</i>	<i>you</i>	<i>your</i>	<i>yours</i>	<i>yourselves</i>
3PL	<i>they</i>	<i>them</i>	<i>their</i>	<i>theirs</i>	<i>themselves</i>

Brown 标记 NN 是单数名词(*candy*, *woman*)的 Brown 标记。Brown 标记集合也区分两种特殊类型的名词:专有名词(专名)和副词性名词。专有名词是一些名字,如 *Mary*, *Smith*, *United States*, 表示特定的人或物。专有名词经常大写,它的标记是 NNP^①。副词性名词(标记是 NR)如 *home*, *west*, *tomorrow*, 用来提供被描述事件周围环境的信息而无需其他修饰语,例如时间或地点。它们和副词的功能类似(见下文)。到目前为止我们提到的标记有如下的复数等价类: NNS(复数名词), NNPS(复数专有名词)和 NRS(复数副词性名词)。所有格以及所属格的扩展: NNS\$(单数名词的所有格), NNS\$(复数名词的所有格), NNP\$(单数专有名词的所有格), NNPS\$(复数专有名词的所有格), NRS\$(副词性名词的所有格)。代词的标记列在表 3.2 中。

3.1.2 名词附属词:限定词和形容词

其他的几种词性标注一般都会伴随着名词出现。限定词描述了名词的特指。限定词的一个子类是冠词。冠词 *the* 表明我们正谈论一些已经知道或可以惟一确定的人或事物。对于 *the tree*, 如果我们早就已经涉及到了这棵树或者所指的上下文已经交代得比较清楚,例如当站在一棵树附近,它体现的指示关系就相当清楚。冠词 *a* (或者 *an*)指的是我们前面没有提到的人或事物。如果我们说 *a tree*, 就表明前面没有提到过这棵树而且它的身份不能通过上下文得到。其他的限定词包括指示代词,例如 *this* 和 *that*。

形容词用来描述名词的属性。下面的斜体部分列出了一些形容词:

a red rose, this long journey, many intelligent children, a very trendy magazine (3.7)

这些修饰名词的用法被称为属性或者修饰词。形容词还有一种表语用法,即用做 *be* 的补语。

The rose is red. The journey will be long. (3.8)

许多语言对于冠词和形容词像名词一样区分了格、数和性别,于是我们可以说冠词和形容词必须与名词对应,即它们必须有相同的格、数和性别。在英语中,正如前面所述,形容词的变形是

^① 实际上,专有名词的 Brown 标记是 NP,但是我们参照 Penn 树库用 NNP 取代 NP,这样 NP 能够保持它在语言学中的常用含义——名词短语(见下文)。我们在以后提及相关标记时也参照 Penn 树库使用两个 N。

在结尾加上派生结尾-ly,或采用比较级形式(*richer, trendier*)以及最高级形式(*richest, trendiest*)。只有一些(主要是比较短的)形容词的比较级和最高级的变形是加后缀-er 或-est,剩下的采用了迂回的形式(*more intelligent, most intelligent*)。迂回形式加入了辅助词,在这里是 *more* 和 *most*。相对于比较级和最高级而言,形容词的基本形式(*rich, trendy, intelligent*)被称为原级、比较级和最高级,说明了应用于名词的形容词所描述属性的程度大小,举例如下:

John is rich, Paul is richer, Mary is richest. (3.9)

Brown 标记 形容词(原级)的 Brown 标记是 JJ,比较级的是 JJR,最高级的是 JJT。还有一种特殊的标记 JJS 表示语义上的最高级,如 *chief, main* 和 *top*。数目是形容词的子类。基数(如 *one, two* 和 6 000 000)的标记是 CD,序数(如 *first, second, tenth* 和 *mid-twentieth*)的标记是 OD。

冠词的 Brown 标记是 AT。单数限定词(如 *this, that*)的标记是 DT;复数限定词(如 *these, those*)的标记是 DTS,可单可复的限定词(如 *some, any*)的标记是 DTI,二者联合的限定词(如 *either, neither*)的标记是 DTX。

数量词是指表达了类似于 *all, many* 和 *some* 的含义的词。限定词 *some* 和 *any* 也可以作为数量词。对应数量词的其他一些词性标注使用标记 ABN(前置数量词: *all, many*)和 PN(名词性代词: *one, something, anything, somebody*)。*there* 用在句首表示存在的时候,它的标记是 EX。

与名词同时出现或者替代名词的最后一类词是疑问代词和限定词,它们用在疑问和关系从句中,标记是 WDT(*wh*-限定词: *what, which*),WPS(所有格 *wh*-代词: *whose*),WPO(宾格 *wh*-代词: *whom, which, that*)和 WPS(主格 *wh*-代词: *who, which, that*)。

3.1.3 动词

动词用来描述动作(*She threw the stone*)、活动(*She walked along the river*)和状态(*I have \$50*)。一个规则的英语动词有下面的一些变形:

- 词根或者原形: *walk*
- 第三人称单数现在时: *walks*
- 动名词或进行时: *walking*
- 过去时或过去分词: *walked*

以上多数形式都有几种用法。原形用于现在时:

I walk. You walk. We walk. You (guys) walk. They walk. (3.10)

第三人称单数有不同的现在时形式:

She walks. He walks. It walks. (3.11)

原形用在带 *to* 的不定式中:

She likes to walk. She has to walk. To walk is fun. (3.12)

还可以用在情态动词之后或者不定式中(不带 *to*)。

She shouldn't walk. She helped me walk. (3.13)

加-ing 的形式用在进行时中(表示一个动作正在进行):

She is walking. She was walking. She will be walking. (3.14)

也可以用做动名词,这是动词获得了名词的一些或者所有属性的一种派生形式。

This is the most vigorous *walking* I've done in a long time. *Walking* is fun. (3.15)

加-*ed* 的形式用做过去时则表明一个动作过去已经发生:

She *walked*. (3.16)

它也可以在现在完成时中用做过去分词:

She *has walked*. (3.17)

以及过去完成时:

She *had walked*. (3.18)

很多动词都是不规则动词,有不同的过去式和过去分词形式。下面是 *drive* 和 *take* 的例子:

a. She *drove* the car. She *has never driven* a car. (3.19)

b. She *took off* on Monday. She *had already taken off* on Monday.

正如名词通常会标记出其特征,如数和格,动词也通常根据其特征进行标记。表 3.3 总结了语言中针对动词的一些语法特征。这些特征可以认为是变形(也称之为合成),例如英语的结尾 -s, -ing, -ed, 或者通过助动词的手段,还有一些在动词群体里伴随动词的词。英语中使用了助动词 *have*, *be* 和 *will* (还有其他的词)来表达观点、语气以及一些时态信息。现在完成时和过去完成时由 *have* 组成,可以从例句(3.17)和例句(3.18)中看到。现在进行时由 *be* 组成,可以从例句(3.14)中看到。使用助动词的形式,相对于英语中过去时的直接变形,可以认为是迂回形式。

表 3.3 动词常见特征

特征类别	例子
主语数	单数和复数
主语人称	第一人称(<i>I walk</i>)、第二人称(<i>you walk</i>)和第三人称(<i>she walks</i>)
时态	现在时、过去时和将来时
体态	进行时和完成时
情态	可能性、虚拟语气和虚拟
分词	现在分词(<i>walking</i>)和过去分词(<i>walked</i>)
语态	主动、被动和中间态

在英语中,有一类有特殊属性的动词:情态助动词,或称情态动词。情态助动词没有普通动词的一些形式(没有不定式,没有进行时),而且在动词群中总是出现在最前面。它们表达了一些情态概念,例如可能性(*may*, *can*)或者责任义务(*should*),如下面的例子所示:

- a. With her abilities, she *can* do whatever she wants to.
 b. He *may* or *may not* come to the meeting. (3.20)
 c. You *should* spend more time with your family.

在英语中,带有助动词 *will* 的将来时形式在各方面与其他情态动词都是等同的:

She *will* come. She *will not* come. (3.21)

Brown 标记 Brown 标记集合使用 VB 来表示动词原形, VBZ 表示第三人称单数(*takes*), VBD 表示过去式(*took*), VBG 表示动名词和现在分词(*taking*), VBN 表示过去分词(*taken*)。情态动词(*can, may, must, could, might, ...*)的标记是 MD。因为 *be, have* 和 *do* 在组成时态和语气时很重要, 所以 Brown 标记对于这些动词的所有形式都有独立的标记。这里将其省略, 但是在表 4.6 中已经列出。

3.1.4 其他词性

副词、介词和小品词

我们已经在变形派生的例子中遇到过副词。副词修饰动词的方式与形容词修饰名词的方式是一样的。副词指定了地点、时间、方式或程度:

- a. She *often* travels to Las Vegas.
- b. She *allegedly* committed perjury. (3.22)
- c. She started her career off *very impressively*.

一些副词, 例如 *often*, 并不是从形容词派生出来的, 也没有后缀 *-ly*。

一些副词也可以修饰形容词, 如例句 (3.23a) 和例句 (3.23b), 或者修饰其他副词, 如例句 (3.23c)。

- a. a *very unlikely* event
- b. a *shockingly* frank exchange (3.23)
- c. She started her career off *very impressively*.

某些副词比较特殊, 例如 *very*。它们修饰形容词和副词, 但并不修饰动词, 我们称之为程度副词。它们的分布与其他副词差别很大, 因此有时候它们被认为是语言中独立的一部分, 称为限定词。

介词最初主要是表示空间关系, 是一些小词 (*small words*):

- in the glass, on the table, over their heads, about an interesting idea,*
concerning your recent invention (3.24)

很多介词都有双重语法功能, 还可以作为小品词。小品词是介词的一个子类, 它们在短语动词结构中 与动词之间存在很强的关系。我们最好是把短语动词看成独立的词汇条目, 它们与派生它们的动词在语法结构和语义属性上都存在着差别。下面举例说明:

- a. The plane *took off* at 8am.
- b. Don't *give in* to him.
- c. It is time to *take on* new responsibilities. (3.25)
- d. He was *put off* by so much rudeness.

这些结构中的动词和介词有时可以分开:

- a. I didn't want to *take* that responsibility *on* right now.
- b. He *put* me *off*. (3.26)

这些短语动词表达专门的特殊意义,并不能从组成它们的动词和小品词直接推断出来。

在某些情况下我们需要知道句子的意思才能区分小品词和介词:*up* 在例句(3.27a)是一个介词,而在例句(3.27b)中则是一个小品词。这个词的意思已经从字面意义“沿一个陡坡向上跑”,见例句(3.27a),转换成越积越多的意思,指“欠下了一个大账单”,见例句(3.27b)。

- a. She ran up a hill.
b. She ran up a bill. (3.27)

Brown 标记 副词的标记是 RB(普通副词: *simply, late, well, little*), RBR(比较级副词: *later, better, less*), RBT(最高级副词: *latest, best, least*), * (not), QL(限定词: *very, too, extremely*)和 QLP(后置限定词: *enough, indeed*)。标注中的下面两种标记同时拥有副词和疑问功能: WQL (*wh-*限定词: *how*)和 WRB(*wh-*副词: *how, when, where*)。

介词的 Brown 标记是 IN,但是小品词的标记是 RP。

连词和补语

剩下的重要词语类别是并列连词和从属连词。并列连词并列连接了两个同类别的词或短语:

- husband *and* wife [名词]
- She bought *or* leased the car. [动词]
- The green triangle *and* the blue square [名词短语]
- She bought her car, *but* she also considered leasing it. [句子]

并列连词的一个功能是连接两个句子(或者从句),正如最后一个例子所示。这也可以用从属连词来实现。在下面的例子中,从属连词用斜体字来表示。

- a. She said *that* he would be late. [命题]
b. She complained *because* he was late. [原因]
c. I won't wait *if* he is late. [条件] (3.28)
d. She thanked him *although* he was late. [让步]
e. She left *before* he arrived. [时间]

类似例句(3.28a)中的从属连词 *that* 或 *for* 的情况用来引入动词的对象,这些对象经常可以被认为是补语。并列连词和从属连词之间的区别是,正如两个术语所表明的,并列连词连接了两个同等的句子,而从属连词把一个从句和主句连接起来。从句通常阐述了一个命题、理由、条件、让步或时间上有关联的一个事件。

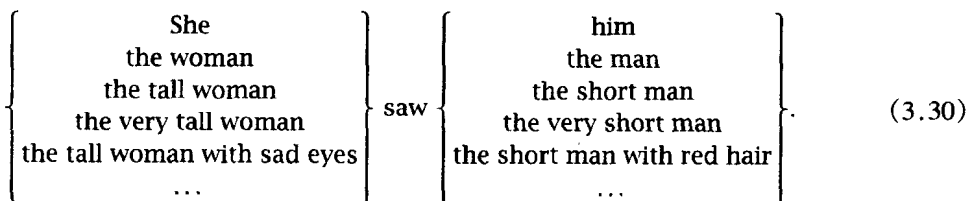
Brown 标记 并列连词的标记是 CC。从属连词的标记是 CS。

3.2 短语结构

语言在词语顺序上有很多的限制,但句子中的词语并不只是简单地作为词性序列而被连接在一起的。不同点在于,词语组成了短语,把一些词聚集成一个语言单位;句法则是研究词语序列和短语结构的规律和约束。

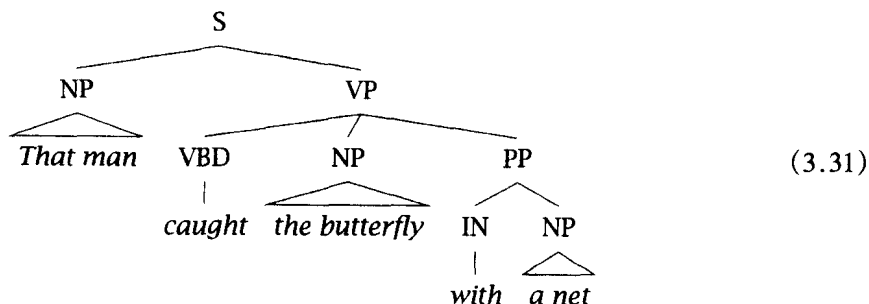
一个基本的思想是,特定的词语集合的行为类似于一个语法成分。语法成分可以通过看它们是否能够出现在不同的位置,并且表现出一致的语法扩展的可能性而检测到。例句(3.29)和例句(3.30)中的例子显示了包含名词及其修饰语的语法成分从位置和短语两个方面进行扩充的证据。

- a. I put *the bagels* in the freezer.
 b. *The bagels*, I put in the freezer. (3.29)
 c. I put in the fridge *the bagels* (that John had given me)



这是 Saussurean 语言学中的范例关系(paradigmatic relationship)概念。在某个特定的句法结构位置(正如上面的名词短语成分),可以互相替换的元素都是一个范例的成员。与之对比的是,如果两个词能够形成类似 *sewed clothes* 或者 *sewed a dress* 的短语或语段,我们就认为它们之间具有语段关系。语段上相关的词的一个重要类别是词语搭配(见第 5 章)。

这一节中将简略地涉及一些主要的短语类型,并介绍语言学家用来模型化短语结构的技术,例如英语句子都有类似下面的典型整体短语结构:



整个句子给定了一个类别 S。一个句子通常可以写成一个主语名词短语和一个动词短语。

名词短语 一个名词经常可以嵌入到一个名词短语(NP)中,名词短语是句子中名词信息聚集起来的句法结构单位。名词是名词短语的中心词,也是决定这个短语语法性质的核心成分。名词短语通常是动词动作的对象,动作的参与者,以及动词描述的活动或状态。它通常含有任意的限定词,零个或者多个形容词短语,中心词(即名词),以及一些后置修饰语,比如介词短语和从句修饰语,这些成分以上述说明的顺序依次出现。名词从句修饰语通常被称为关联从句。下面这个名词短语说明了这些可能的情况:

The homeless old man in the park that I tried to help yesterday (3.32)

介词短语 介词短语(PP)以介词开始并且包含一个名词短语补语。它们可以出现在其他所有主要的短语类型中。特别是在名词短语和动词短语中,它们通常用来表示空间、时间、位置以及其他属性。

动词短语 类似于名词短语以名词为中心词的方式,动词是动词短语(VP)的中心词。通常,动词短语在句法结构上依靠动词来组织起句子的所有元素(动词短语不包含的主语名词短语除外)。下面是一些动词短语的例子:

- a. *Getting to school on time* was a struggle.
- b. He *was trying to keep his temper*. (3.33)
- c. That woman *quickly showed me the way to hide*.

形容词短语 复杂形容词短语(AP)并不常见,包含这些短语的例子如下面句子中粗体字所示:She is **very sure of herself**; He seemed a man who was **quite certain to succeed**.

3.2.1 短语结构语法

句子的句法结构分析告诉我们如何从词语的意思来确定句子的含义。例如,它可以告诉我们在句子所描述的事件中谁对谁做了些什么。比较:

Mary gave Peter a book. (3.34)

Peter gave Mary a book. (3.35)

例句(3.34)和例句(3.35)使用了同样的词,但是却有着不同的含义。在第一个句子中,书是由 Mary 给 Peter 的,而在第二个句子中却是由 Peter 给 Mary 的。正是词语顺序使得我们可以区分出施动者和受动者。

其实很多语言,比如拉丁语和俄语,都允许句子以不同的方式组织词语顺序却不改变句意,而采用格标记来指明施动者和受动者。这种类型的语言称为自由词语顺序语言,意味着词语顺序并不用来指明谁是施动者,而主要用来指明论述结构。其他的语言类似于英语,对于句子中哪些词语可以移动的程度规定得相对更严格一些。在英语中,最基本的词语顺序是主-谓-宾:

The children (subject) *should* (auxiliary verb) *eat spinach* (object). (3.36)

通常来说,修改词语顺序只是为了表达特殊的语调类型。在疑问句(或者是问题)中,主语和第一个助动词通常是颠倒的:

Should (auxiliary verb) *the children* (subject) *eat spinach* (object)? (3.37)

如果叙述中没有涉及到助动词,do 的一种形式将会出现在句首位置(*Did he cry?*)。在祈使句(命令或者要求)中没有主语(可以推断出涉及到的人):

Eat spinach! (3.38)

相对于疑问句和祈使句而言,基本的句子称为陈述句。

词语顺序规则经常被表述成重写规则。一个重写规则的形式为“类别→类别”,表明左边的符号序列可以重写为右边的符号序列。要生成语言的一个句子,我们以(句子)初始符‘S’开始。下面有一个简单的重写规则集合:

S	→	NP VP	AT	→	<i>the</i>	
NP	→	$\begin{Bmatrix} \text{AT NNS} \\ \text{AT NN} \\ \text{NP PP} \end{Bmatrix}$	NNS	→	$\begin{Bmatrix} \textit{children} \\ \textit{students} \\ \textit{mountains} \end{Bmatrix}$	
VP	→	$\begin{Bmatrix} \text{VP PP} \\ \text{VBD} \\ \text{VBD NP} \end{Bmatrix}$	VBD	→	$\begin{Bmatrix} \textit{slept} \\ \textit{ate} \\ \textit{saw} \end{Bmatrix}$	(3.39)
PP	→	IN NP	IN	→	$\begin{Bmatrix} \textit{in} \\ \textit{of} \end{Bmatrix}$	
			NN	→	<i>cake</i>	

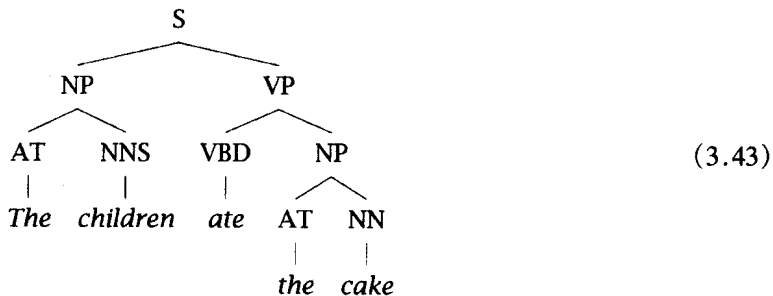
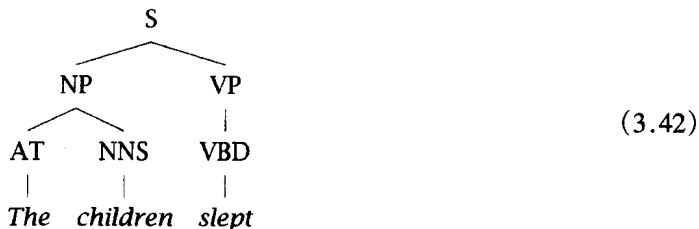
右边的规则把前面章节引入的句法结构类别(标注符号)重写为对应类别的一个词。这部分语法通常会被分离出来成为一个词典。这些规则的本质是,一个特定的句法结构类别可以被重写为一个或多个其他的句法类别或者词语。重写的概率只依赖于类别,并不依赖于任何周围的上下文,因此这种短语结构语法通常被称为上下文无关文法。

我们用这些规则可以派生出句子。例句(3.40)和例句(3.41)的派生是两个简单的例子。

S
→ NP VP
→ AT NNS VBD
→ *The children slept* (3.40)

S
→ NP VP
→ AT NNS VBD NP
→ AT NNS VBD AT NN
→ *The children ate the cake* (3.41)

表示短语结构的最直观的方法是树。树的叶子节点作为终结节点,内部节点作为非终结节点。在这样的树中,每个非终结节点和它的直接子节点或者局部树对应着重写规则的一个应用。子节点的顺序生成了句子中词语的顺序,而且树只有一个根节点,是语法的开始符。分析树(3.42)和分析树(3.43)对应例句(3.40)和例句(3.41)的派生。树的每个节点表示我们假设的成分。



最后一种表示成分的方法是加(带有标记的)括号。括号集合划出了各成分并且通过加标记表示了非终结符的类别。分析树(3.43)加上带标记的括号后如例句(3.44)所示:

[S [NP [AT The] [NNS children]] [VP [VBD ate] [NP [AT the] [NN cake]]]] (3.44)

大部分形式化自然语言语法的重写规则共有的一个属性就是递归:也就是说,重写规则可以被使用任意多次。在我们给出的示例语法中,一个 PP 可以包含一个 NP,而 NP 又可以包含另一个 PP。因此我们可以得到递归的扩展,如图 3.1 所示。这里,介词短语序列由多次循环使用重写规则生成:“NP→NP PP; PP→IN NP”。这个派生使用了两次上述循环,但是我们可以应用 3 次,4 次或 100 次。

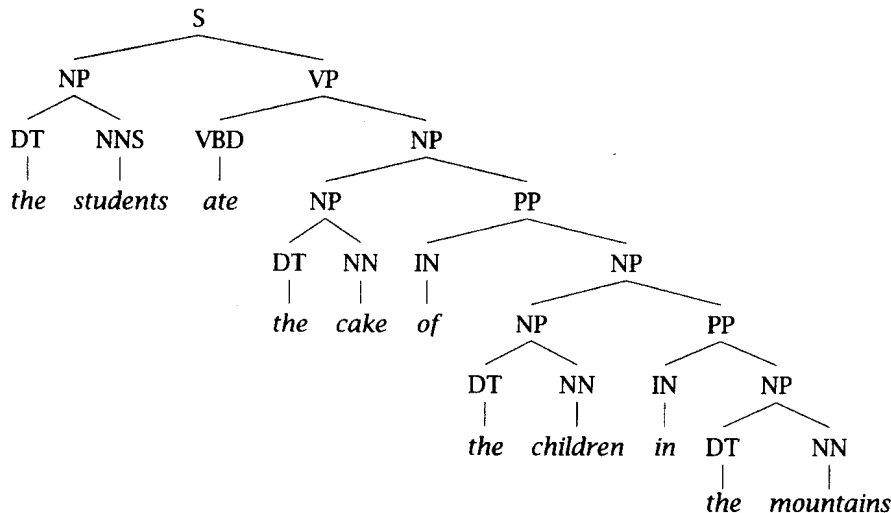


图 3.1 一个递归短语结构扩展的例子

递归使得类似 VP 或者 NP 的单个非终结符扩展为大的词语结构。例如,图 3.1 中符号 VP 被扩展到 9 个词: *ate the cake of the children in the mountain*。一种情况是,在句子的派生过程中,由普通重写规则生成并且语法上联系在一起的两个词语能够被插入其中的词语分开,这类现象称为非局部依存性,因为两个词是语法上相互依存的,即使它们在句子中相隔很远。

依存性显示出局部性的一个例子是主语-动词呼应,一个句子中的主语和动词在数和人称上是对应的。我们有 *She walks*, *He walks*, *It walks*, 相对于 *I walk*, *You walk*, *We walk*, *They walk*。也就是说,以-s 结尾的动词表明了第三人称单数,当且仅当主语也是第三人称单数。主语和动词互相呼应,即使它们之间被插入了其他的词或者短语,类似下面的例子:

The women who found the wallet were given a reward. (3.45)

如果我们只看直接邻居,那么似乎只能说 *the wallet was*。只有通过对这个句子进行完整的语法分析才能知道 *the women* 是主语而且 *be* 的形式只能是复数。

另一种重要的非局部依存关系是远距离依存关系,例如 *wh*-抽取^①。例如例句(3.46b)中

① 在语言学著作中,词组“远距离依存关系”一般指超出了三元语法模型的范围。我们把这种效果描述成“非局部依存关系”,并且保留了语言学中词组“远距离依存关系”常用的依存含义,这个含义是指这种依存关系能够在短语结构树中跨过任意数目的节点。

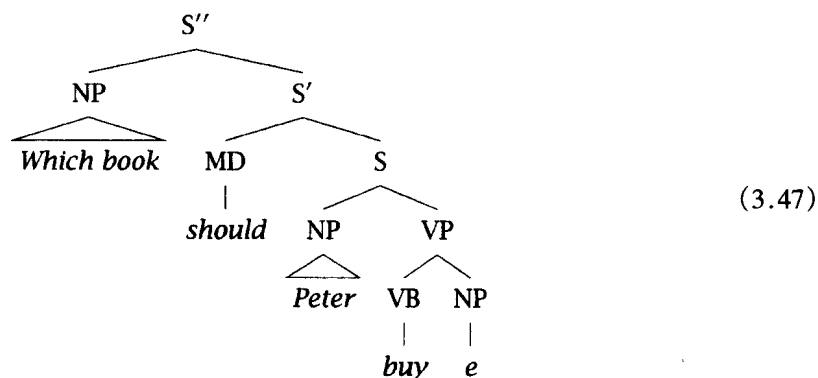
的短语 *which book* 移到了句子最前面。

- a. Should Peter buy a book?
b. Which book should Peter buy? (3.46)

没有任何移位理论的分析,就可以明显看到在 *buy* 和 *which book* 之间有远距离的依存关系,否则我们就不能说 *book* 是 *buy* 的对象。

- ▼ 非局部现象对于很多统计 NLP 方法都是一个挑战,比如描述局部依存关系的 n 元语法模型。一个 n 元语法模型会把例句(3.45)中 *wallet* 后面的词预测为 *was* 而不是 *were*。这些问题将在第 11 章中进行深入讨论。

很多版本的短语结构语法的最后一个特征是空节点。当一个非终结符被重写为空时会出现空节点。例如,注意到我们可以说 *Eat the cake!* 而不需要主语,于是建议加入一条规则 $NP \rightarrow \emptyset$ 。一个 NP 非终结符因此可以被写为空。这一般通过在树的节点下面插入一个 $\emptyset$ 或者 e 来表示。使用这种符号,例句(3.46b)可以写成如分析树(3.47)所示的结构:



这里所提出的简单短语结构模型采用了上下文无关语言的观点。例如,一旦我们把‘VP’扩展到 VBD NP,继而就可以扩展到 *sewed* NP,我们可以用任意名词短语来代替 NP。在我们决定怎样扩展 NP 时,动词 *sewed* 提供的上下文是不可用的。这种上下文的不可用性是上下文无关文法的关键属性。我们可以把 VP 扩展到一个很自然的类似 *sewed clothes* 的短语,但是也可能选择一个无意义的类似 *sewed wood blocks* 的扩展。

- ▼ 怎样包含必要的依存关系是概率分析的一个中心议题,将在第 12 章中讨论。

3.2.2 依存关系:论元和附属语

另一个重要的组织概念是依存关系。在下面的句子中:

Sue watched the man at the next table. (3.48)

Sue 和 *the man* 是观察事件的依赖词,我们可以说动词 *watch* 有两个论元。介词短语 PP *at the next table* 是 *man* 的依赖词,它修饰 *man*。

名词短语常常作为动词的论元。动词的论元可以被描述为不同的层次。我们可以通过语义角色来将论元分类。动作的施动者是做事情的人或物,受动者是接受某件事物的人或物,其他的角色,例如工具或者目的,描述了语义关系的其他类别。另一种方法是,我们可以根据语

法关系描述论元的句法可能性。所有的英语动词都会有一个主语,即出现在动词前面的名词短语。许多动词都会带一个名词短语作为宾语,它经常直接出现在动词后面。代词在作为动词的主语时使用主格,在作为动词的宾语时使用宾格。在例句(3.49)中, *children* 是 *eat* 的主语(*children* 是动作 *eat* 的施动者),而 *sweet candy* 是 *eat* 的宾语(*sweet candy* 是被操作的对象,动作的受动者):

Children eat sweet candy. (3.49)

注意 *candy* 并没有变形。在英语中,代名词是惟一在作为宾语的时候改变形式的名词。

一些动词在其后面带两个名词短语作为宾语,都是宾语形式:

She gave him the book. (3.50)

在这个句子里面, *him* 是间接宾语(描述接受者,间接得到事物的人), *the book* 是直接宾语(描述受动者)。其他类似的动词还有“送出”(*sending*)类的动词和“联系”(*communication*)类的动词。

a. She sent her mother the book.

(3.51)

b. She emailed him the letter.

这些动词还允许它们的对象有另一种表示形式,即接受者出现在介词短语中:

She sent the book to her mother. (3.52)

具有格标记的语言通常区分了这些 NP,并且以宾格表示受动者,以给予格(dative)表示接受者。

在语义角色和语法功能间通常都有一定的关系,例如施动者通常是主语,但是也有一些例外。在 *Bill received a package from the mailman* 中, *mailman* 是施动者。语义角色和语法功能之间的关系也会随着语气改变而改变(前面没有讨论表 3.3 中的这个特征)。许多语言区分了主动语态和被动语态(或者简单主动和被动)。主动对应表达一个动词论元的默认方式:施动者是主语,受动者是宾语:

Children eat sweet candy. (3.53)

在被动语态中,受动者成为了主语,施动者降格为一个间接角色。在英语中这意味着两个论元的顺序颠倒了,而且主语用介词短语 *by*-短语来表示。被动语态由助动词 *be* 和过去分词组成:

Candy is eaten by children. (3.54)

在其他的语言中,被动语态转换仅仅只涉及到格标记的变化以及一些动词的变形。

子类化

我们已经看到,不同的动词和与之联系的实体(人、动物、事物)的数目是不一样的。一种不同在于及物动词和不及物动词的对比。及物动词有一个(直接)宾语,不及物动词则没有:

a. She brought a bottle of whiskey.

b. She walked (along the river).

(3.55)

在例句(3.55a)中, *a bottle of whiskey* 是 *brought* 的宾语。我们不能在使用 *bring* 时不带宾语。动词 *walk* 是不及物动词的一个范例。在例句(3.55b)中没有宾语。但是,其中句尾的介词短语表达了该活动的位置。

语法学家们努力对动词的依赖词进行分类。他们找到的第一个区别存在于论元和附属语

之间。主语、宾语和直接宾语都是论元。一般来说,论元表达了参与动词活动的中心实体。大多数论元都用 NP 表达,但是它们也可以表达成 PP, VP 或者从句。

- a. We deprived him *of food*.
b. John knows *that he is losing*. (3.56)

论元可以被划分成主语以及其他非主语论元,这些非主语论元统称为补语。

附属语是那些与动词联系不是很紧密的短语。它们一般是可选的,与此相对的是很多补语都是必需的(例如, *bring* 的宾语是必需的)。附属语相对于补语而言更容易移动。下面的有关附属语的例子告诉了我们动词所描述的动作或状态的时间、地点或方式,如:

- a. She saw a Woody Allen movie *yesterday*.
b. She saw a Woody Allen movie *in Paris*.
c. She saw the Woody Allen movie *with great interest*. (3.57)
d. She saw a Woody Allen movie *with a couple of friends*.

从句(句子内部的句子)也可以作为附属语或者子类化论元,能够表达与动词之间的一些关系。在例句(3.28)中,(a)包含一个论元从句,其他的是附属语。

有时候,很难区别附属语和补语。介词短语 *on the table* 在第一个句子中是补语(它被 *put* 子类化了,而且不能被省略),在第二个句子中是一个附属语(可以被省略):

She put the book *on the table*. (3.58)

He gave his presentation *on the stage*. (3.59)

传统的论元/附属语之间的差别确实反映了传统语言学中的绝对性基础。但是在很多例子中,我们会发现一些中间状态,如下例所示:

- a. I straightened the nail *with a hammer*.
b. He will retire *in Florida*. (3.60)

我们并不清楚上面斜体的 PP 是否应该被认为涉及到了动词描述事件的核心。在统计语言学方法中,考虑动词和依赖词之间关联的程度可能会更有意义。

我们依据动词允许子类化的补语类型来将动词分类。换句话说,动词依据特殊的补语子类化。例如, *bring* 子类化为可带一个宾词。这里有一个子类化论元列表以及例句。

- 主语。The children eat candy.
- 宾语。The children eat *candy*.
- 介词短语。She put the book *on the table*.
- 形容词述语。We made the man *angry*.
- 不带 to 不定式。She helped me *walk*.
- 带 to 不定式。She likes *to walk*.
- 分词短语。She stopped *singing that tune* eventually.
- that 从句。She thinks *that it will rain tomorrow*. that 通常可以省略: She thinks *it will rain tomorrow*.

- 疑问形式从句。She is wondering *why it is raining in August*. She asked me *what book I was reading*.

虽然大多数的补语都是我们已经见过的短语单位,例如 NP 和 AP,但是最后的条目却不是,因为它们是比较 S 更大的单位。从句 *why it is raining in August* 包含一个整句 *it is raining in August* 以及前面的一个补充成分。这么大的一个从句被认为是一个 S' (读成 S Bar) 成分。关联从句和主从句也可被分析成 S' 的成分。

通常动词有几种可能的论元模式。一个可以伴随动词出现的特殊论元集合称为子类框架。这里有一些英语中常见的子类框架。

- 不及物动词。NP[subject]. The woman walked.
- 及物动词。NP[subject], NP[object]. John loves Mary.
- 带双宾语的动词。NP[subject], NP[direct object], NP[indirect object]. Mary gave Peter flowers.
- 带 PP 的不及物动词。NP[subject], PP. I rent in Paddington.
- 带 PP 的及物动词。NP[subject], NP[object], PP. She put the book on the table.
- 句子补语。NP[subject], clause. I know (that) she likes you.
- 带句子补语的及物动词。NP[subject], NP[object], clause. She told me that Gary is coming on Tuesday.

子类化框架体现了补语的句法规律。也有很多的语义规律被称为选择约束或者选择倾向。例如,动词 *bark* 趋向于以 *dog* 作为主语。动词 *eat* 趋向于以可以吃的东西作为宾语。

The Chihuahua barked all night. (3.61)

I eat vegetables every day. (3.62)

违背选择倾向原则的句子会显得很奇怪。

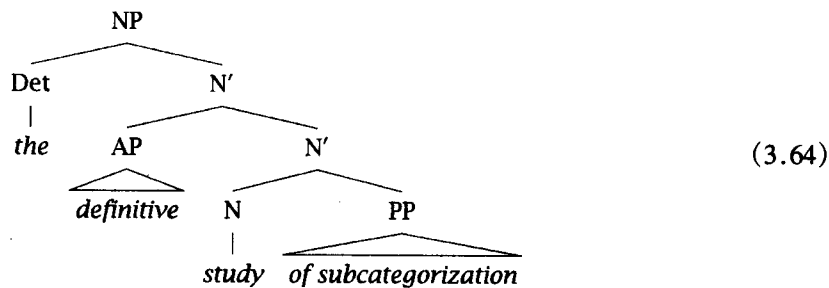
- a. The cat barked all night.
b. I eat philosophy every day. (3.63)

▼ 我们将会在 8.4 节中详细讨论选择倾向。

3.2.3 X'理论

上面提到的短语结构规则在自然语言的短语构成中并不具有系统性,句子中不同依赖词的出现规则也是这样。但是,现代语法强调有很多这样的规则。一个重要的思想是短语有一个中心词。我们讨论名词短语和介词短语的原因是,它们是分别包含了名词和介词及其所有依赖词的成分。名词或介词作为短语的中心词^①。语言学家们还认为,短语中依赖词分布在中心词周围的方式有很大的系统性。一个中心词以及它的补语形成了一个小成分。这个成分可以被附属语修饰成为一个较大的成分,最终这个成分会与一个专有词、主语或者限定词联合起来形成一个最大的短语。下面的分析树(3.64)给出了一个一般的形式。

^① 但是,通常描述的动词短语又有些异常,因为它们包含了动词的所有补语,却不包括主语。



中间成分被称为 N' 节点(读为“N bar 节点”)。这基本上是两层 bar 思想(我们认为 XP 是 X”),但是因为附属语修饰语在 N' 层可以递归出现,所以使得这种思想显得很复杂。这也表明了名词可以有任意多个形容词短语修饰语。有时人们使用更多或者更少层 bar 的思想。

虽然有词语顺序的不同,但是这种通用的成分模式在短语类型之间重复使用。这种思想被称为 X' 思想,其中 X 可以用来代表一个跨词语类别的变量。

3.2.4 短语结构歧义

现在我们可以使用重写规则来生成句子。在句法分析中它们会更经常地被使用,句法分析是指给出一个特殊的词语序列,重构它的派生或者短语结构树的过程。我们把根据句子构建的一棵短语结构树称为一个分析。例如,分析树(3.43)是例句(3.41)的一个句法分析。

在大多数情况下,对于一个特定的词语序列,可以给出多个不同的短语结构树。一个基于英语中完全语法的句法分析器通常能找到一个句子的上百个分析。这种现象称为短语结构歧义或者句法结构歧义。在绪论中我们看到了一个句法结构歧义句子的例子,即例句(1.10): *Our company is training workers*。一种经常出现的句法结构歧义是附着歧义。

附着歧义发生在可以被两个不同节点生成的短语中。例如,对于例句(3.39)中的语法,有两种方式可以生成例句(3.65)中的介词短语 *with a spoon*:

The children ate the cake with a spoon. (3.65)

它可以作为动词短语的一个子节点生成,如图 3.2(a)中的分析树所示,或者作为一个名词短语的子节点生成,如图 3.2(b)中的分析树所示。

不同的附着有着不同的含义。与动词短语的高层附着使勺成为一个小孩在吃蛋糕时使用的工具。与名词短语的低层附着告诉我们被吃的是哪块蛋糕(带着一个勺的蛋糕,而不是有糖霜的蛋糕)。因此,解决附着歧义对于找到正确的语义解释是很重要的。

句法结构歧义中的一个被深入研究过的子类是 Garden Pathing 现象。一个 *garden path* 句子可以把你引到一条死胡同里。例如,句子中会出现一些看上去不应该在那里的多余词:

The horse raced past the barn fell. (3.66)

(Bever 1970)中的例句(3.66)可能是 Garden Pathing 句子中最著名的一个例子。在大多数人遇到词 *barn* 的时候,他们会简单地依据 *The horse ran past the barn* 的意思来构建一个句法分析。但是附加词 *fell* 无法再加入到分析中。我们只好回溯到 *raced*,再构造一个完全不同的分析,对应意思 *The horse fell after it had been raced past the barn*。Garden Pathing 是一种进入可疑分析以后不得不回溯尝试重新构造正确分析的现象。

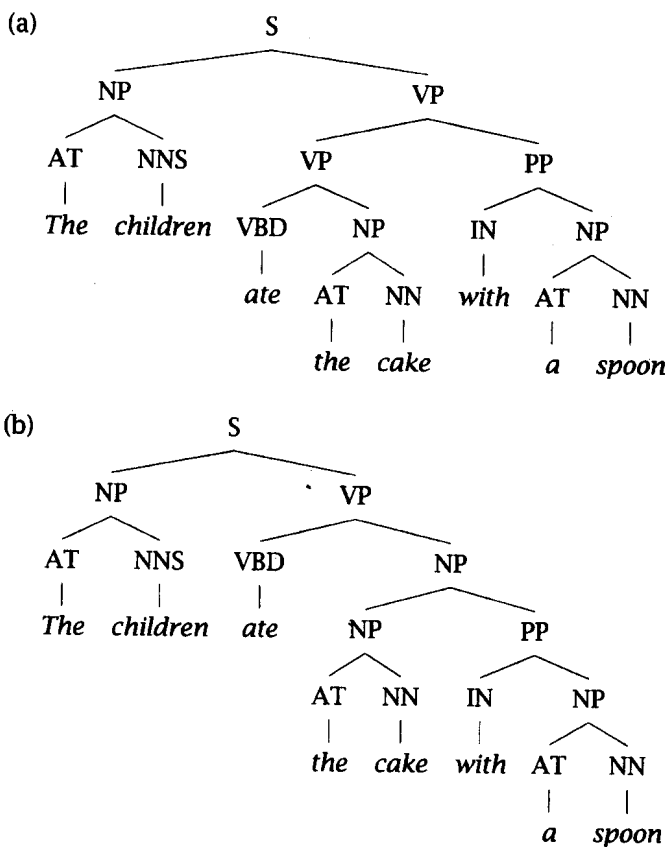


图 3.2 一个介词短语附着歧义的例子

Garden Pathing 句子在口语中并不会造成任何问题。语义倾向、说话者们遵循的交流原则以及语调模式通常会阻止我们进入 Garden Pathing, 参见 (MacDonald et al. 1994; Tanenhaus and Trueswell 1995)。例句 (3.66) 中 *horse* 和 *raced* 之间的语气停顿告诉听众 *raced* 将会引导一个简化的关系从句, 而不是主句的动词。但是, 在阅读书面英语中的复杂句子时 Garden Pathing 仍然是一个现实问题。

我们已经看到了因为语法歧义出现不只一个句法分析的例句。大部分句子都是这种类型的, 但是也有可能一个句子根本就找不到对应的句法分析。原因可能是在语法中根本就没有包括句子生成所用的规则。另一种可能是这个句子不合乎语法, 或者语法结构本身不完整, 下面我们就给出一个不合乎语法的句子。

*Slept children the. (3.67)

区别“不合乎语法”与“语义异常”这两种概念很有必要。例如, 类似下面的句子看起来是很奇怪的, 导致它们很不和谐的原因是语义解释不连贯, 然而例句 (3.67) 根本就无法解释。

- a. Colorless green ideas sleep furiously. (3.68)
- b. The cat barked.

人们通常使用 # 标记来指明语义、语用和文化怪异, 区别于前面引入的语法结构异常的符号。

3.3 语义和语用

语义研究的是词语的含义、结构和说话的方式。语义可以分成两部分:研究单个词的语义(即词义)以及单个词的含义是怎样联合起来组成句子(或者是更大的单位)的含义的。

研究词义的一条途径是研究词义之间是怎样互相联系的。我们可以把词语组织成为词汇层次,例如,在词网(WordNet)中定义了上位词和下位词。上位词(hyponymy 或 hyponymy^①)是指一个意义宽泛的词,譬如动物(*animal*)是猫(*cat*)的上位词。下位词(hyponymy)则具有更加特指的含义:猫(*cat*)是动物(*animal*)的一个下位词。一般来说,如果 w^1 是 w^2 的上位词,那么 w^2 是 w^1 的下位词。反义词(antonym)是指具有相反意义的词,例如 *hot*(热)和 *cold*(冷), *long*(长)和 *short*(短)。部分-整体关系称为 meronymy。*tire*(轮胎)是 *car*(小轿车)的一个 meronymy, *leaf*(叶子)是 *tree*(树)的 meronymy。整体-部分的关系称为 holonym。

同义词(synonym)是指具有相同意义的词(或者具有相近意义的词), *car* 和 *automobile* 是同义词。同形异义词(homonym)是指写法相同,但是(概念上或者历史上)有着不同含义,看上去丝毫不相关的两个词。例如 *suit*(意思是 *lawsuit* 或 *set of garment*)和 *bank*(意思是 *river bank* 或者 *financial institution*)。如果词的多个意思(或意义)有联系,那么我们称之为多义词(polyseme)。词 *branch* 是一个多义词,因为它的多个意思(植物的自然分支,或者一个分离的但是仍然是中心组织的依赖部分)是相关的。词汇歧义会涉及到同形异义词和多义词。在同形异义词中,如果两个词不仅有同样的写法,还有同样的发音,则称为同音异义词(homophony)。因此词 *bass* 在指一种鱼或者一种低音时是同形异义词,但不是同音异义词。

▼ 语义消歧将会在第7章中重点讨论。

一旦我们得到了单个词的含义,就需要把它们汇集成整个句子的意思。这是比较困难的,因为自然语言经常不遵守总的含义可以由各部分的含义严格推导出来的组成原则。词 *white* 在下列短语中涉及到了十分不同的颜色:

white paper, white hair, white skin, white wine (3.69)

白头发是灰色的,白皮肤是带点红色的,白酒实际上是黄色的(但是“黄酒”听起来就不那么吸引了)。*white hair*, *white skin* 和 *white wine* 这个集合可以作为词语搭配的例子。整体含义是指各部分的含义之和加上一些不能从各部分推导出来的额外的语义信息。

▼ 第5章将重点讲述词语搭配。

如果词语含义与短语含义之间的关系很远,那么我们称这个短语为成语。例如,短语 *kick the bucket* 描述了将死(*dying*)的过程,与 *kicking* 或者 *buckets* 都没有关系。我们或许可以解释这个短语的历史起源,但是在现今的语言中它是完全非合成(non-compositional)的。另一个例子是名词-名词复合词 *carriage return*,它指的是标记一行结尾的字符。很多年轻人并不知道它的最初含义:当打字机换到新行时会把它的架子重置到页左边的位置。

在组合更大的语义单位时会遇到其他更多更重要的问题,我们这里不做详细讨论。例如,

^① 后者是正规形式,前者是常用形式。

作用范围(scope)的问题。数量词和操作的作用范围可以超越一个或多个短语或者句子。在下面的句子中,我们可以把数量词 *everyone* 的范围定在否定词 *not* 上(意指没有一个人去看电影),或者把否定范围定在数量词上(意指至少有一人没有去看电影):

Everyone didn't go to the movie. (3.70)

为了推导出以上句子的正确含义,我们需要根据上下文来确定哪个解释是正确的。

在词和句子之后的下一个大的单位是篇章(discourse)。关于篇章的研究可以阐明文本中句子之间隐含的关系。在一段叙述性的篇章中,我们可以找出下面的句子描述的是一个例子,细节还是重复说明或其他内容。在谈话中,我们需要确定涉及的说话行为类型以及次序之间的关系(问题、叙述、要求或声明等)。篇章分析中的一个核心问题是指代关系消解。

a. Mary helped *Peter* get out of the cab. *He* thanked her.

b. Mary helped *the other passenger* out of the cab. *The man* had asked her to help him because of his foot injury. (3.71)

指代关系发生在涉及到同一个人或物的多个名词短语之间。例句(3.71a)中的 *Peter* 和 *He* 以及例句(3.71b)中的 *the other passenger* 和 *The man* 指的是同一个人。指代消解在信息抽取中是很重要的。在信息抽取中,我们扫描一个文本,寻找特定类型的事件,诸如自然灾害,恐怖袭击或者公司收益。其任务是识别出事件中的参与者以及其他事件典型信息(例如公司兼并中的收购价格)。要做好这项工作,正确地辨认指代关系对于跟踪参与者是非常关键的。

Hurricane Hugo destroyed 20,000 Florida homes. At an estimated cost of one billion dollars, the disaster has been the most costly in the state's history. (3.72)

如果我们把上述短文(3.72)中的 *Hurricane Hugo* 和 *the disaster* 确认为同一个实体,就可以把 *Hugo* 作为下面问题的答案: *Which hurricane caused more than a billion dollars worth of damage?*

篇章分析是语用论研究的一个部分,语用论主要研究世界知识和语言习惯是怎样与字面含义相互影响的。指代关系是一种语用现象,因为它们受世界知识的约束。例如,要解决短文(3.72)中的指代关系,就必须知道 *hurricane* 是一种 *disaster*。语用论研究领域还没有受到统计语言学的太多关注,不仅因为很难用统计手段为世界知识的复杂性建模,还因为训练数据的缺乏。但是,有两个领域逐渐受到更多的关注,一个是指代关系的消解,另一个是为对话中的语言行为建模。

3.4 其他研究领域

传统上语言学被划分成语音学、音韵学、形态学、句法学、语义学以及语用学。语音学主要研究语言的发音,诸如谐音、元音和语调等。音韵学主要研究语言发音系统的结构。语音学和音韵学在语音识别和语音合成中非常重要,但是因为我们并没有涉及到语音,因此本书中并不具体讨论这方面的内容。在我们首次涉及语音学和音韵学时将会介绍少量基本概念。

除了研究那些处理不同语言层次的领域外,还有一些特殊的语言学研究领域。社会语言学研究社会组织和语言之间的相互影响。语言随时代的改变是历史语言学的主题。语言分类学主要研究语言对语言学工具的不同使用,以及它们是怎样基于所用工具的方式被分成不同

类的。语言获取调查了孩子们是怎样学习语言的。心理语言学家专注于实时语言的产生和语言理解的问题,以及语言展现在脑海里的方式。许多领域都很有可能会用到定量方法。数理语言学习惯于实现一些使用非定量数学思想的方法。

3.5 深入阅读

读者如果希望深入了解一些语言学领域,可以参阅(Newmeyer 1988)。在许多领域中,我们都可以感觉到统计语言学的影响,比如语料库的广泛使用。统计语言学采用的是定量的方法。

(De Saussure 1962)是结构语言学中的一个里程碑。非语言学家如果要对语言学领域有深入了解,可以参阅 Cambridge Encyclopedia of Language (Crystal 1987)。还可以参阅非常流行的一本书(Pinker 1994)。Marchand (1969)提供了对英语中的词语派生概率的极其全面的研究。Quirk et al. (1985)全面地概括了英语的语法。最后,查找语法结构词汇(还有其他一些变形和语义)方面比较好的参考资料是(Trask 1993)。

对于语音识别和语音合成的一个好的介绍是(Waibel and Lee 1990; Rabiner and Juang 1993; Jelinek 1997)。

3.6 习题

3.1 [★]标出下面段落中的词语的词性。

The lemon is an essential cooking ingredient. Its sharply fragrant juice and tangy rind is added to sweet and savory dishes in every cuisine. This enchanting book, written by cookbook author John Smith, offers a wonderful array of recipes celebrating this internationally popular, intensely flavored fruit. (3.73)

3.2 [★]给出 5 个名-名复合词的例子。

3.3 [★]识别出下面的句子的主语、直接宾语和间接宾语。

He baked her an apple pie. (3.74)

3.4 [★]下面两个句子的含义有什么差别?

- a. Mary defended her.
- b. Mary defended herself. (3.75)

3.5 [★]英语句子的标注词序列是什么? (a) 陈述句; (b) 祈使句; (c) 疑问句。

3.6 [★]下列形容词和副词的比较级和最高级是什么?

good, well, effective, big, curious, bad (3.76)

3.7 [★]给出下列动词的原形、第三人称单数形式、过去式、过去分词以及现在分词。

throw, do, laugh, change, carry, bring, dream (3.77)

3.8 [★]将下列句子改写成被动语态。

- a. Mary carried the suitcase up the stairs.
b. Mary gave John the suitcase. (3.78)

3.9 [★]介词和小品词之间的差别是什么? 下列句子中的 *in* 有什么语法功能?

- a. Mary lives in London.
b. When did Mary move in?
c. She puts in a lot of hours at work. (3.79)
d. She put the document in the wrong folder.

3.10 [★]分别给出 3 个及物动词和不及物动词的例子。

3.11 [★]补语和附属语之间的差别是什么? 下列句子中的斜体短语是补语还是附属语? 是什么类型的补语和附属语?

- a. She goes to Church *on Sundays*.
b. She went *to London*.
c. Peter relies *on Mary* for help with his homework. (3.80)
d. The book is lying *on the table*.
e. She watched him *with a telescope*.

3.12 [★]下列句子中的斜体短语是附着歧义的例子。两种可能的解释是什么?

Mary saw the man *with the telescope*. (3.81)

The company experienced growth in classified advertising *and preprinted inserts*. (3.82)

3.13 [★]下列短语是复合的还是非复合的?

to beat around the bush, to eat an orange, to kick butt, to twist somebody's arm, help desk, computer program, desktop publishing, book publishing, the publishing industry (3.83)

3.14 [★]动词短语是复合的还是非复合的?

3.15 [★]在下面的句子中, *a few actors* 或者 *everybody* 都可以覆盖整个句子。其含义的差别是什么?

A few actors are liked by everybody. (3.84)

第4章 基于语料库的工作

本章在开始从事基于语料库的工作之前,首先给出一些简单的建议。统计自然语言处理的主要需求包括计算机、语料库和软件。计算机和语料库自身的许多方面变化得非常快,没有必要把注意力放在它们上面。而且,在很多情况下,即使你的计算机或者语料库在各方面并不是最理想的,也只能局限在当前条件下工作。至于软件,本书并不打算教你编程技巧,而且,如果有人对书中描述的算法有兴趣,那么他可以立刻使用一种编程语言来编程实现它。即便如此,我们也会提到一些有用的编程语言和工具。

本章接下来将讲述一些有趣的主题,诸如语料格式以及在处理生语料(电子版未经过标注的文本)时遇到的问题。正式的研究工作开始之前,需要对文本做低级处理,这个步骤非常重要,但常常被忽略。我们会看到,词语和句子的准确定义存在很多困难,实践中通常用一些并不完美的启发式方法来解决这些困难。由于这些方法不精确,所以会影响后续的结果,这一点非常重要,一定要时刻牢记在心里。

最后,本章论述了数据标注问题。用程序为文本加上明确的标记(但通常是由人来完成的,以此来指明文本的结构以及语义。标记用处很大,但是在标记的种类和内容方面,也引起了一些问题。我们将介绍 SGML 标记(以及 XML 标记)的入门知识,接着将转向各个独立的主题,例如标注某种语言的语料时如何选取标注集。

4.1 基础知识

4.1.1 计算机

文本语料库通常都比较大,处理大量的文本需要相当多的计算资源。在早期的处理中,这是限制语料库运用的主要原因。例如,很早以前构造 Brown 语料库时(20 世纪 60 年代),仅仅把语料库中的所有词排序形成一个词表就需要 17 个小时的专门处理,这是因为当时所用的计算机(IBM 7070)的内存仅仅有 40 KB,所以排序算法必须把已经排序好的数据存进磁带设备里。今天,我们使用一台普通的计算机,可以在几分钟之内做完同样规模的排序。

统计自然语言处理方法不仅需要大量的空间来存储语料,而且经常需要从语料库中收集大量的统计信息,所以要求计算机有比较快的存取速度。这就需要一台硬盘足够大,内存足够多的计算机。在当今飞速发展的时代,最有意义的事情莫过于对硬盘容量的需求。幸运的是,所有的变化都在向好的方向发展,我们通常需要的就是这样一台比较好的计算机,内存可以便宜地进行扩充(然而,仅仅是几年前,要装备一台速度足够快,内存和硬盘空间足够大的计算机,仍然需要花费一大笔经费)。

4.1.2 语料库

表 4.1 列出了一些主要的分发文本语料库的机构,这些机构的工作都基于语言学研究的

目的。大多数机构使用这些语料库都要支付适当的费用^①。如果你的预算没有包括购买语料库这笔花费,网上有许多免费的文本资源,范围从电子邮件、网页到许多书和杂志。这些免费资源并不是标注好的语料,但是有现成的工具可以为这些文本自动加上比较好的标记。无论如何,能够给原始文本加上标记,这本身就是一件具有挑战性的工作。网上还可以找到其他的在线文本资源。

表 4.1 主要语料库的供应者及其 URL

Linguistic Data Consortium (LDC)	http://www ldc upenn edu
European Language Resources Association (ELRA)	http://www icp grenet fr/ELRA/
International Computer Archive of Modern English (ICAME)	http://nora hd uib no/icame.html
Oxford Text Archive (OTA)	http://ota ahds ac uk/
Child Language Data Exchange System (CHILDES)	http://childes psy cmu edu/

工作中涉及到语料库时,我们不得不担心统计分析结果的有效性。一个语料库是按照某种标准收集的特殊文本材料。例如,Brown 语料库是专门为 1961 年使用的美国书面英语设计的典型样本语料库,参见(Francis and Kučera 1982:5~6)。在构造过程中引入了一些标准,如各种文本在数量上应和实际出版物成比例,另外还剔除了诗句,因为“诗句引入了特殊的语言学问题”。

因此,在 Brown 语料库中得出的结论并不符合英国英语或者美国口语。例如,2.2.7 节估计出来的英语的熵严重依赖于用于估计的语料库。通常认为诗歌的熵比其他书面文本的熵要高,因为诗歌可以不必理会语义甚至语法的要求。所以 Brown 语料库对于评定诗歌的熵没有什么帮助。再举一个更通俗的例子(参见第 16 章)。在文本分类中,一个系统的性能会由于时间关系极度恶化,因为某一时刻抽取出来用于训练的样本经过一年或者两年之后就失去了原本的代表性。

语料库是不是人们感兴趣的典型样本是这里的主要问题。如果我们从一个样本中得出的结论同样适用于一般对象,就可以说这个样本是典型的。我们并不准备讨论确定典型样本的方法,因为这个问题在关于语料库语言学的著作中已经有了详细的说明。为了创建平衡语料库,建议读者阅读这些文献。所谓平衡语料库是按照事先确定好的某种重要标准,把每个子类的文本按照一定比例收集到语料库中。在统计自然语言处理中,通常把感兴趣的某个领域的大量数据收集起来作为语料库,不管这些数据是怎么构造的。在这种情况下,训练文本的数量要比文本的平衡性有用得多,通常我们只简单地使用可以获得的所有文本。

总之,并没有一种简单的方法来评定一个语料库是否有代表性,但是在进行统计自然语言处理时一定要记住这个问题的重要性。选择语料库或者发布结果的时候,至少要明白,语料库包含哪种类型的文本,得到的结果是否可以移植到其他感兴趣的领域。

▼ 10.3.2 节将讨论语料库的多变性对于词性标记准确性的影响。

4.1.3 软件

有许多程序可以用来查看文本语料和分析语料数据。然而,我们建议读者自己写程序处

^① 价格非常昂贵,但是对于研究和非营利目的使用的机构,通常每张 CD 的价格在 100~2000 美元的范围内,这样的价格也反映出在收集和处理语料过程中需要做的大量工作。

理数据,这样,真正需要的软件就是简单的文本编辑器和某种编程语言的编译器或者解释器。除此之外,我们可能还会经常用到其他工具软件,例如在语料库中进行搜索的软件。稍后将简单地介绍一些这样的工具。

文本编辑器

我们需要一个普通文本编辑器来查看文件中的内容。标准而且便宜的选择是 UNIX 平台(或者 Windows 平台)下的 Emacs, Windows 平台下的 TextPad, 或者 Macintosh 平台下的 BBEdit。

正则表达式

对于很多领域、很多程序和编辑器,都有在文本中匹配某种模式的需求,这比通常的连续字符匹配要复杂得多。用于这种匹配的最普遍的符号是正则表达式,正则表达式可以描述正则语言的模式,正则语言是一种可以被有限状态自动机识别的语言。如果你对正则表达式还不是很熟悉,应该慢慢熟悉它。正则表达式可以用在许多普通文本编辑器里(Emacs, TextPad, Nisus, BBEdit, 等等),可以和许多工具(例如 grep 和 sed)一起使用,可以在许多程序设计语言中(例如 Perl, C, 等等),或者内置在语言中,或者作为该语言的一个库。在(Hopcroft and Ullman 1979), (Sipser 1996)和(Friedl 1997)中可以找到正则表达式的介绍。

编程语言

当前的大多数统计自然语言处理使用 C/C++。对于大量数据集合和文本的处理来说,用 C/C++ 这样的语言进行编码可以获得很高的效率。但是对一些辅助的文本处理来说,使用其他语言可以更加节省人力资源。很多人使用 Perl 来进行文本的预处理和重新改变格式。这种语言的语法中内置的正则表达式具有相当强大的功能。通常,用解释型的语言来做这种类型的工作比所有代码都用 C 语言写要快一些。更早之前,可能不用 Perl 而用 awk,尽管用 awk 有更多的限制。Python 是另一种选择,它受到传统程序员的青睐,但是在 Python 里用正则表达式没有 Perl 里面用起来容易。本书作者之一还在很大程度上使用了 PROLOG 语言。内置的数据库功能和能够方便地处理复杂的数据结构的特点,使得 Prolog 在某些方面表现得相当优秀。但是,它同样也缺少 Perl 的方便处理正则表达式的功能。除此之外,还有其他的专门为文本计算设计的语言,例如 SNOBOL/SPITBOL 和 Icon,这些语言在人文学科计算领域相当受欢迎,但是还没有渗透到统计自然语言处理领域。最近几年,Java 兴起了,尽管 Java 在速度上没有 C 快,但是它有其他许多明显的优点,比如面向对象的特点,自动内存管理机制,以及许多现成的有价值的库。

编程技巧

这个小节并不是一般计算机算法的替代,我们仅仅给出一些有用的技巧。

文字编码 统计自然语言处理通常都要处理大量的文字,然而 C 或者 C++ 这样的编程语言只提供了相当有限的功能来处理文字。在统计自然语言处理和信息抽取中,输入文字时,都采用把文字映射到数字的方法(当需要输出的时候才映射回文字)。这种方法具有许多优点,因为比较数字非常容易,也非常快。我们也把文字的标记映射到它所属的类型,每种类型都有一个单独的数字。有许多不同的方法可以实现这种功能。一种比较好的方法是维护一个大的哈希

表(哈希函数可以把一组对象映射到一个特定范围的整数,例如 $[0, \dots, 127]$)。使用哈希表可以很容易地查出一个文字以前是否出现过,如果出现过,还可以返回它所对应的数字;否则就把这个文字添加到哈希表中,并且为它赋一个新值。这里使用的数字可以作为一个文字数组的索引(如果把文字个数限制到 65 000 或者更少,效率会特别高,因为可以用 16 比特数字来保存这些数字),或者把它们当做哈希表里存储的规范形式字符串的地址。输出的时候相当方便,不需要从数字转化为文字,字符串就可以被打印出来。

还有其他的有用数据结构,例如各种各样的树。读者可以参考一本有关算法的书,例如(Cormen et al. 1990)或者(Frakes and Baeza-Yates 1992)。

收集计数信息 很多统计自然语言处理的首要工作是收集各种观察对象的出现次数,以此作为估计概率的基础。最直接的方法是建立一个比较大的数据结构(数组或者任何其他类似的结构),然后在这个结构中对感兴趣的事件进行计数。这种方法在实践中效果通常特别差,因为这种方法需要大量可同时随机存取的内存空间。除非你的计算机有足够的内存可以容纳这些结构,否则这种方法或者由于内存交换过多而意外结束,或者越来越慢而趋于停滞。对于这种数据收集的程序来说,比较好的方法是首先为每个出现的对象指定一个标记,然后用程序对这些标记排序,最后统计标记出现的次数。事实上,这些后续的步骤都可以使用已有的工具来完成(例如 UNIX 系统中的 sort 和 uniq)。通过本书配套网站可以获取 CMU-Cambridge 统计语言模型工具包,这个工具包就成功地使用了类似的策略。

4.2 文本

文本通常有两种形式:生文本和标注文本。“标注”这个术语是指把某种分类代码插入到一个计算机文件中,这种分类代码通常并不是文件的组成部分,但是通过这些分类代码,我们可以了解文件的结构或格式信息。几乎所有处理文本的计算机系统都使用了某种分类标记。商业字处理软件也使用标记,只不过它隐藏了这些标记,给用户所见即所得的感觉。通常,用统计自然语言处理方法处理语料时,我们给可以见到的部分做出明确的标记。这也是语言学家的首要工具是普通文本编辑器的部分原因。

人类语言中的许多特点使得文本自动处理相当困难,即使低水平的自动处理也很困难。这里我们将讨论一些应该了解的基本问题。这些讨论将主要涉及英文文本中最基本的问题,但并非只涉及英文。

4.2.1 低级格式问题

垃圾格式(或内容)

由于语料库的来源复杂,语料中可能存在无法处理的各种各样的格式或内容,它们是没有用处的,需要过滤掉。这些无用格式或者内容包括:文档页眉和分隔符、排版代码、表和图表,计算机文件中的无用数据,等等。如果数据来自 OCR(光学字符识别),OCR 处理可能会引入一些问题,比如页眉、页脚和插入的内容(表,图和脚注)破坏了文本的段落结构。通常还会有识别错误等问题。如果你的程序用来处理连续的英语文本,那么诸如表和图片之类的其他内容都被认为是无用的。在进一步处理之前需要一个过滤器来过滤掉这些无用的内容。

大小写

原始 Brown 语料库都是大写字母(在原来的源文件中,在字母前加一个 * 号表示大写字母)。全部大写的文本近年来已经很少见了,但即使是现代文本,也有一个如何对待大写的问題。特别是,如果两个词除了某些字母的大小写之外完全相同,我们该不该认为它们是相同的呢?考虑到许多原因,例如我们仅仅想研究一下定冠词的用法或者名词短语结构,这种时候,对于 *the*, *The* 和 *THE*, 可以认为它们是一样的。把词语全部变成大写或者小写非常容易,但问題是我们同时还想保留 *Richard Brown* 和 *brown paint* 中的两种截然不同的 *Brown*。在很多情况下,保留这种不同可以很容易地正确区分出来姓名,但是有时候也无法做到。一种启发式方法是,把每个句子开头的大写字母(英语中开头的词通常第一个字母都是大写的)转成小写字母,把一串连续大写的词当做标题和副标题,这样其余的大写字母词语就可以认为是名字,它们的大写字母可以保留。这种启发式方法非常有效,但是,自然而然它也存在一些问题。第一个问題就是必须能够正确地识别出句子的结尾,但这并不总是那么容易的,我们稍后将讨论这个问題。某种语言流派(比如 *Winnie the Pooh*)为了强调某个词语的重要性,会把这个词写成大写,但这并不表明它是一个名字。无论如何,这种启发式方法会错误地把处于句子开头的人名和全部都是大写字母的句子中的人名转成小写的。这种由于源文件引发的错误通常可以被容忍(因为普通词语比人名要常见得多),但是有时候会对估计结果造成较大的偏差。通过维护一张正确的人名列表(有可能还有更多的信息,例如如何命名人名、地名和公司名)可以取得较好的结果。但是,通常来说,并没有一种简单的方法可以正确地检测出人名。

4.2.2 标记化:什么算是一个词

通常,文本处理的早期工作是把输入文本切分成称为词次(token)的单位,词次可以是一个词,一个数字或者是一个标点符号。这个过程称为标记化。处理标点有不同的方法。通常我们希望保留句子的边界(参见 4.2.4 节),但是句子内部的标点经常被过滤掉,这种方法不是很好。近来已经重视了标点中蕴含的信息,标点符号标得即使再差,逗号和破折号也可以给出文本中宏结构的线索,并且可以看出成分修饰关系。

在语言学中,到底什么是词,这是一个很难回答的问题。语言学家们通常把答案总结为:在不同的层次上有不同的词,例如音位词和语义词,不同的层次没有必要有完全相同的定义。低调的计算语言学家怎么看待这个问题呢? Kučera and Francis (1967)提出了一个实用的图表词表示法,定义如下:前后有空格的连续字母组成的字符串,可以包含连字符和省略号,但是不能包含其他的标点符号。遗憾的是,生活从来都不是这么简单的,即使对于寻找一个切实可行的定义来说也是这样。Kučera 和 Francis 应该是按照直觉给出了定义,在实践中发现,数字和像 \$ 22.50 这样的词并不是严格遵守上面的定义的。事情比想像得要复杂很多,如果使用新闻组和网页等在线资源作为数据,情形更加严重,即使局限于新闻专线,还是有各种奇怪的符号可以推测为词语,例如 *Micro \$oft*, 网络公司 *C|net*, 还有各种各样的用标点符号组成的笑脸,例如:-)。即使不理睬这些自创词,识别词也是一件相当困难的工作。英语中主要的线索是空格,如词语之间的一个空格,或者一个制表符,或者一个换行符,但即使是这种标记也不总是可以信赖的。那么,主要的问题在哪里呢?

句点

词语前后也不总是存在空格。标点符号经常紧跟在词语后面,比如逗号、分号和句点。乍看起来,去掉词语后面的标点似乎很容易,但是对于句点来说却存在问题。大多数句点的作用是表明句子结束,其他情况表示缩写,例如 *etc.* 或者 *Calif.*,这些作为缩写的句点应该保留在词语中作为词语的一部分。有时候把这些作为缩写的句点保留下来相当重要,因为通过这个句点可以帮助我们区分 *Washington* 的缩写 *Wash.* 和首字母大写的动词 *Wash*。特别需要注意的是,当一个类似 *etc.* 的缩写词出现在句子末尾的时候,句子末尾只保留一个句点,这个句点可以同时表示句点的两种意思。本段前面出现的 *Calif.* 就是一个例子。在词法意义上,这种现象称为缩略。4.2.4 节将进一步讨论如何判别表示句子结尾的标点符号。

单撇号

英语中的 *I'll* 或者 *isn't* 这类缩写是一个让人头痛的问题。按照上面的定义应该把它们看做一个图表词,但许多人凭直觉强烈地感觉到是两个词,只不过是两个词的缩写形式罢了,是 *I will* 和 *is not* 的缩写。一些处理程序(和一些语料库,例如 Penn 树库)把这种缩写切分为两个词,另一些则不进行切分,不切分会存在一些问题。传统的第一个语法规则是:

$$S \rightarrow NP VP$$

遇到句子中的缩写,诸如 *I'm right* 的时候,很明显,这个规则就不再适用了。另一方面,如果切分了,那么语料中就会存在像 *'s* 和 *n't* 这样的词。

dog's 和 *child's* 这样的短语如果不是 *dog is* 或者 *dog has* 的缩写,通常会被认为是 *dog* 的所有格形式。但是 3.1.1 节曾经提到过,当 *'s* 在名词短语中跟在其他成分之后作为接语时,上面的假设是不对的,例如在英语中, *The house I rented yesterday's garden is really big*。考虑到这种情况,到底是把 *dog's* 看成一个词还是两个词再一次变得很矛盾。Penn 树库再一次选择了后者,也就是看成两个词。处理词末尾的单引号是一个相当棘手的问题,通常它代表一对引号的结束,所以不应该是词的一部分;但是如果它是跟着一个 *s* 出现,比如 *boys' toys*,可能表示一个(不发音的)复数所有格,所以又应该是词的一部分,前提是其他的所有格也是这么处理的。总之,对于一个词法分析器来说,没有一种简单的方法可以确定到底是上述各种情况中的哪一种。

连字符: 不同形式表示相同的词

处理输入中的连字符可能是最困难的领域之一。中间带有连字符的一串字母应该看成一个词还是两个? 同样,这个问题的答案是有时候一个,有时候两个。这反映了文本中连字符来源的多样性。

一个来源是排版印刷。把词语分开,中间插入连字符以改进版面的对齐,这是传统的做法。从原始的排版印刷品中抽取的数据存在这种行分割的连字符。这种连字符看起来好像很好处理,只需要找到一行中最后的连字符,丢弃它,并把本行词的一部分和下一行连接起来即可。但是这里同样有缩略的问题。如果有另一个不是这种情况的连字符,那么这个连字符之后的位置是切分文本的合理位置,这种情况下只有一个连字符出现,而不是两个。所以删掉一行末尾的连字符并不总是对的,通常很难分辨哪个连字符用于行分割。

即使这种行分割连字符不存在(真正的电子文本中一般不出现),还是存在需要解决的难题。一些带有连字符的字符串很明显应该被看成一个词,例如 *e-mail*, 或者 *co-operate*, 或者 *A-l-plus* (在 *A-l-plus commercial paper* 中的一种财务等级证书)。虽然通常把它们当成一个词,例如 *non-lawyer*, *pro-Arab* 和 *so-called*,但在许多情况下存在较大的争议。这里的连字符应该称为词汇连字符,它们通常被插入到小构词要素之前或者之后,有时候充当分裂元音序列的分隔符。

第三类连字符用来帮助区分正确的词组。*A common copy-editing practice* 就是用连字符连接一个复合的前置修饰语。看下面的例子:

- a. the once-quiet study of superconductivity
 - b. a tough regime of business-conduct rules
 - c. the aluminum-export ban
 - d. a text-based medium
- (4.1)

连字符也出现在其他地方,可以在某种程度上表示引用的短语或者一些数量和比率。

- a. the idea of a child-as-required-yuppie-possession must be motivating them
 - b. a final "take-it-or-leave-it" offer
 - c. the 90-cent-an-hour raise
 - d. the 26-year-old
- (4.2)

在这些情况中,我们大致把连字符连接起来的一串词当成一个词。在很多语料库中,这种类型的连字词出现得特别普遍,这将极大地增加词汇表的容量(大多数是词典之外的条目)。同时,如果这些条目不被切分,也会使文本的语义结构变得更加难以理解^①。

这个领域的一个特殊问题是许多情况下连字符的使用非常不一致,一些文本和权威著作中使用 *cooperate*, 然而另一些使用 *co-operate*。另一个例子是在 Dow Jones 新闻专线中,可以找到 *database*, *data-base* 和 *data base* (第一个和第三个比较常见,第一个在计算机软件中出现得比较多,第三个用于公司资产领域,在使用中它们没有任何词法上的区别)。写作本章初稿的时候,作者(偶然)使用了 *markup*, *mark-up* 和 *mark(ed) up* 这三种形式。细心的编辑会发现这个问题并且要求前后使用的词语要一致,但是我们使用的很多文本并没有交给细心的编辑进行核对。而且,无论如何,我们通常会使用来源不同,采用不同约定的文本。这就意味着总会存在同一个词的多种形式,可能在有些文本中把它们当成一个词,而有些则当成两个。对于这种情况,最好的方法是只有一个词位(词典中一个词条拥有单一的意义)。

最后,英国印刷惯例是在破折号和词语之间添加空格,而美国印刷惯例是直接跟着词语的长破折号。在计算机文件中,有时这个破折号会被认为是一个特殊字符或者是多个破折号,传统计算机字符集的限制意味着有时候可能把它当成一个连字符,这又将进一步增加上面问题的难度。

① 一种可行方案是把词切分开,但是同时加上标记,这些标记可以表明这些词原来是用连字符连接起来的,本章后面将要提到这种方法,这样就不会丢失信息了。

相同形式表示不同的“词语”

不同序列的字符串代表同一个词,建议大体上忽略差别,将其当成同一个词。反过来,同样的字符序列可能代表不同的词,这一点也非常重要。同形异义词就有这样的问题,两个词位有着相同的形式,例如作为名词的 *saw* 是一种工具,但是 *saw* 也是动词 *see* 的过去时。在这种情况下,需要把 *saw* 放到两个不同的词位中去。

▼ 自动处理这个问题的方法将在第7章中详细讨论。

其他语言中的分词

许多语言中词和词之间没有空格,所以遇到空格就切分的基本分词算法在这些语言中一点用处都没有。这些语言包括大多数东亚语言,例如中文、日文和泰文。古希腊人手写的古希腊语也是没有空格的。空格(和重音符号)都是后人加上去的。在这些语言中,分词是一个更加重要和富有挑战性的任务。

德语中大多数的词语有空格分割,但是复合名词都写成单独的词,例如 *Lebensversicherungsgesellschaftsangestellter*,指人身保险公司雇员。把复合词当成一个词在许多方面具有语言学上的意义,至少从音位上来说是这样的。但是对于处理来说,我们希望把这种复合词切分开,或者至少了解它的内在结构,这就变成了一个限制性的分词任务。上面这种现象虽然不是一个规则,但在英语中也存在这样的复合词,特别是当它普遍到拥有一个独特意义的时候。前面提到过的 *data base* 和 *database* 就是这种情况。*hard disk* 比较常见,但在计算机出版物中也会发现有 *harddisk*,这也是一个例子。

非词语分界的空格

到目前为止,我们的主要问题就是如何区分不是由空格分界的字符序列。反过来,把分开的词合到一起也是一个问题。在这种情况下,词语被空格分开,但是 we 想把它们当成一个单独的词语。一种可能的情况是德语中复合词问题的相反问题。如果决定把 *database* 当成一个单独的词,那么即使写成 *data base*,也应该把它当成一个词。更加普遍的情况是电话号码,我们想把 9365 1873 当成一个单独的“词”,对于像 *New York* 或者 *San Francisco* 的多词地名,也希望将其当成一个词。如果这个问题和连字符问题同时出现,问题将变得特别困难,就像下面这样:

the New York-New Haven railroad (4.3)

这里的连字符并不表示把相邻的两个词连成一个词,如果真把 *York-New* 当成一个语义单位,就犯了个大错误。

在语言学中比较让人感兴趣的还有其他一些问题。出于许多考虑,希望把动词短语(*make up*, *work out*)的习惯搭配当成一个词位(参见 3.1.4 节)。但是这个问题很难处理,因为很多情况下小品词和动词是分开的(*I couldn't work the answer out*),所以识别动词短语的习惯搭配通常都留到后续程序中处理。我们也希望能把其他的固定搭配当成一个词位,例如 *in spite of*, *in order to*, *because of*,词法分析程序通常可以把它们当成一个单独的词。LOB 语料库使用了类似的策略,*because of* 这样成对的词都被标记为单独的词性,这里的介词使用了所谓

的同上(ditto)标签来标记这种词性。

某种语义类型信息的不同编码

许多读者可能觉得前面提到的电话号码的例子没有那么复杂,因为他们自己的电话号码都写成像 812-4374 这样的形式,或者其他类似的形式。事实上,即使你不打算处理多语言文本,只要你的程序需要处理多国文本或者不同风格的文本,那么它必须能够处理印刷体的差别。特别是电话号码,很明显属于一种语义类别,存在许多不同的出现形式。我们从“The Economist”杂志的某期广告中挑选出一些不同国家的各种格式的电话号码,如表 4.2 所示。不同的电话号码使用了不同的空格、句点、连字符、括号甚至斜杠,用不同的方式对数字进行分组,即使在同一个国家,表示方式也经常是不同的。另外,电话号码中可能还包含国际或者国内的长途区号,或者两种区号都包括(见表中的前三个 UK 条目),或者仅仅显示一个当地号码。可以用括号、加号等标点符号来显式或隐式地表示这些不同的部分。如何处理这种包含无限格式的问题是信息提取领域的一个标准问题。通常多数做法是手工构造正则表达式来匹配这些格式,但是这种方法比较脆弱。通过学习语义格式来构建自动方法是一种比较有意义的做法。

表 4.2 “The Economist”杂志的某期中出现的不同格式的电话号码

电话号码	国家	电话号码	国家
0171 378 0647	UK	+45 43 48 60 60	Denmark
(44.171) 830 1007	UK	95-51-279648	Pakistan
+44 (0) 1225 753678	UK	+411/284 3797	Switzerland
01256 468551	UK	(94-1) 866854	Sri Lanka
(202) 522-2230	USA	+49 69 136-2 98 05	Germany
1-925-225-3000	USA	33 1 34 43 32 26	France
212. 995.5402	USA	++31-20-5200161	The Netherlands

▼ 本书中并没有大规模地介绍信息提取,但是在 10.6.2 节中会有进一步的讨论。

语音语料库

我们的讨论都集中在书面文本,语音记录语料库也具有额外的挑战性。语音语料通常包含更多习惯用语,它使用各种类型的语音表示来显示发音区别,包含许多句子片段,也包含像 *er* 和 *um* 这样的填充语。例句(4.4)来源于从 LDC 得到的 Switchboard 语料库,是语音记录的一个典型摘录。

Also I [cough] not convinced that the, at least the kind of people that I
work with, I'm not convinced that that's really, uh, doing much for the (4.4)
progr-, for the, uh, drug problem.

4.2.3 词法

是否应该把 *sit*, *sits* 和 *sat* 这种类型的词区别对待也是一个值得考虑问题。这个问题和大写字母问题比较类似,但是传统上认为它在语言学上更有意义。首先,感觉上把所有形式都当成一个词位是自然而然的。通常文献上称词干化,也就是一个去除词缀只留词干的过程。换

句话说,这个过程就是把(文中的词)按照曲折变化形式(或异体形式)进行归类,试图找到曲折形式的词条或者词位。词干化意味着词位水平上的消歧,例如使用 *lying* 的时候,到底是表示“卧倒”的动词 *lie-lay*,还是表示“欺骗”的 *lie-lied*。

信息抽取领域广泛的实验研究表明,当性能评价标准是查询平均值(Salton 1989; Hull 1996)的时候,词干化对于经典 IR 系统的性能提高没有帮助。词干化总会大幅度地提高一部分查询的效率,但是同时也会使另一部分查询的效率下降。从语言学角度来看,这是一个有点让人吃惊的结果,因此搞清楚这种现象的原因就显得非常重要。主要原因有三个。

第一,把一个词干的各种形式看成一个词位看起来很好,但是经常会浪费一些信息。例如,在 *Bill is operating a tractor* (见 3.1.3 节)这句话中, *operating* 是动词 *operate* 的一种时态变形,通常也用于偏正关系的名词短语中,例如 *operating system* 或者 *operating costs*。不难看出,如果搜索 *operating system*,在变形词上直接进行搜索要比在所有包含 *operat* 和 *system* 的段落中搜索好得多。考虑另一个例子,某人输入了 *business*,词干化将返回所有包含 *busy* 的文档,这不可能是一个好的结果。

第二,词法分析把一个词次切分成几个词次。然而,把密切相关的信息组合到一起也是值得的,尽管这将减少词汇量。在各种不同的统计自然语言处理领域中,实践中经常把多字词组看成一个特殊的词次,这样做可以改进系统的性能。屈折词是有用的并且大小合适的语块。

第三,大多数信息抽取研究是在英语上做的,尽管近来多种语言工作有逐渐增加的趋势。英语中只有很少的词态,所以智能化处理词态并不是很有必要。许多其他语言拥有非常丰富的变形和派生系统,所以对于词法分析的需求很迫切。这些语言的完全形式词典,就是分别列出所有的词的所有形式的词典,它肯定会非常大。例如,班图语(中部非洲和南部非洲使用)有丰富的动词变形。下面是从 KiHaya(坦桑尼亚)语来的一个例子。主语的前缀和宾语一致,还有时态:

akabimúha
a-ka-bi-mú-ha
1SG-PAST-3PL-3SG-give
'I gave them to him.'

(4.5)

由于历史原因,班图语的一些拼写法在很多词素之间都有空格。但是,在连接拼写法的语言中,必须进行词法分析,因为这些语言中存在一个广泛体系,在动词词根之前可以出现代词和时态标注,在词根之后还可以出现许多其他的词素,形成一个巨大的组合。芬兰语(Finnish)以每个动词都有百万种变化形式而闻名。

似乎可以从以上段落中得到一个结论,在具有丰富词素的语言中,可以通过拆开变形后的词素而不是派生出新的词素来获得感兴趣的内容。但是,这个假设需要在有大量变形词法的语言中对感兴趣的问题仔细进行测试。

从信息检索得来的结果并不需要应用到某一个或者全部的统计自然语言处理应用中去,认识到这一点很重要。它甚至不需要应用于所有的信息检索。词法分析也许在其他应用中更加有用。词干化对于信息检索系统中无相互影响的评估不起作用。在这种系统中,提交一个查询并进行处理,但没有进一步的输入,通过对返回的一组文件计算准确性来得到评估结果。然而,在相互作用的上下文环境信息检索中,语形学规则分析有重要的作用。计算机不在意奇特的词干,比如 *business* 的词干 *busy*,但是人们会在意。当 *business* 词干化为 *busy*,并且返回一个含有 *busy* 的文件时,人们不知道下一步该做什么。而且,也没有关于人机交互是否影响词

干化的系统性研究。这种研究将会非常有用, *saw* 就是一个例子(你想把它抽取成 *see*, 而不是“切割行动”), 或者有些情况下想获得词干(从 *arbitrariness* 得到 *arbitrary*), 但是有些情况下不想(从 *business* 得到 *busy*)。由于需要人工输入, 所以在统计自然语言处理工作中, 知识缺乏条件下的自动词干化存在很大的困难。

▼ 第 15 章将进一步讨论词干化和信息检索。

4.2.4 句子

什么是句子

关于“什么是句子”, 第一个答案是, 以一个“.”, “?”或者“!”结尾的内容。我们已经提到过, 只有一部分句点的功能是表示句子的结尾, 另外一些句点用来表示缩写或者同时具有两种功能。大体上 90% 的句点是句子分界的标志, 参见(Riley 1989), 但是, 得出这个基本的启发经过了相当漫长的过程。我们还要注意另外一些容易被忽视的问题。有时候一些标点符号会把原本被认为是一个整句的句子分隔开。在冒号、分号和破折号这些标点符号前面、后面或者中间的部分经常会被认为是一个句子。例如下面这个句子中的“:”:

The scene is written with a combination of unbridled passion and sure-handed control: In the exchanges of the three characters and the rise and fall of emotions, Mr. Weller has captured the heartbreaking inexorability of separation. (4.6)

与此相关的一个事实是, 有时候句子并不按照一定的次序排列, 而是以一种奇怪的方式排列着。正常排列的部分往往不会是句子本身, 而是从句。这种类别可以从有直接引语引用的情况中得到, 例如下面的句子:

“You remind me,” she remarked, “of your mother.” (4.7)

直接引语存在第二个问题, 将引号放到句子的最后一个标点后面是标准的排版惯例(特别是在北美)。因此, 上面的例句的结束并不是以句点标识的, 而是以句点后面的引号标识的。

图 4.1 粗略地显示了上面的论述所提到的启发式句子分割算法的实质。实际上, 大多数系统都采用了这种类型的启发式算法。由于在开发中投入了足够的人力和物力, 所以这些系统工作得非常好, 至少在构建系统时处理特定文本领域工作得非常好。但是任何此类解决方法都会在词法处理程序的其他部分遇到和启发式处理同样的问题。构造词法分析器通常要求开发者亲手写大量的代码, 并且要有这个领域的专业知识, 而且分析器通常都比较脆弱, 领域相关性比较强。

近来, 对于强原则性句子边界检测算法的研究增加了不少。Riley(1989)使用了统计学分类树来确定句子的边界。分类树使用的特征包括句点前面和后面出现的词的大小写形式和长度, 还有不同词出现在句子边界前后的先验概率(需要大量标注好的训练语料才可以计算出这些概率)。Palmer and Hearst(1994; 1997)通过简单地使用前后词的词性分布避免了获取上述概率数据的困难, 使用神经网络方法来预测句子的边界, 构造了一个健壮的、很大程度上语言无关的高效率(大约 98% ~ 99% 的正确率)边界检测算法。Reynar and Ratnaparkhi(1997)以及(Mikheev 1998)开发了一个针对此类问题的最大熵系统, 在句子边界预测上可以达到 99.25%

的精确率^①。

- 在 . ? ! (和可能的 ; : -) 出现位置之后加一个假设的句子边界。
- 如果假设边界后面有引号, 那么把假设边界移到引号后面。
- 除去以下情况中句点的边界资格:
 - 如果在句点之前是一个不总出现在句子末尾的众所周知的缩写形式, 而且通常后面会跟一个大写的名字, 例如 *Prof.* 或者 *vs.*。
 - 如果句点前面是一个众所周知的缩写形式, 但是句点后面没有大写词。这样即可正确地处理像 *etc.* 或者 *Jr.* 这样的大多数缩写用法, 这些缩写一般出现在句子的中间或者末尾。
- 如果下面的条件成立, 则除去 ? 或者 ! 的边界资格:
 - 这些符号后面跟着一个小写字母 (或者一个已知名字)。
- 认为其他假设边界就是句子的边界。

图 4.1 句子边界的启发式检测算法

- ▼ 检测句子边界可以看成是一个分类问题, 我们将在第 16 章中讨论分类问题, 例如分类树和最大熵模型。

句子的长度

在语言学分类中, 做传统的计算语言处理实验的时候, 句子通常比较短。部分原因是许多传统上使用的分析工具的运行时间和句子长度成指数关系。由于这个限制, 无法处理 12 个字或者超过 12 个字的句子。但是, 需要注意的是, 许多文本类别的典型句子都相当长。新闻专线的最常见的句子长度是大约 23 个词。表 4.3 是一篇新闻专线文本中的句子长度数据的图表。

表 4.3 新闻专线文本中的句子长度。“百分比”列显示每一范围的百分比, “累计百分比”列显示了某一长度以下的累计百分比

长度	数量	百分比	累计百分比
1~5	1317	3.13	3.13
6~10	3215	7.64	10.77
11~15	5906	14.03	24.80
16~20	7206	17.12	41.92
21~25	7350	17.46	59.38
26~30	6281	14.92	74.30
31~35	4740	11.26	85.56
36~40	2826	6.71	92.26
41~45	1606	3.82	96.10
46~50	858	2.04	98.14
51~100	780	1.85	99.99
101+	6	0.01	100.00

^① 8.1 节将给出精确率这个技术术语的定义和相关讨论, 但是它的定义和人们的直观理解是相符的, 它是正确分类的比率。

4.3 数据标注

我们可以在普通的文本语料库上做许多工作,然而,在已经有文本结构的语料上,我们通常更容易学到更多东西。可以用手工、自动或者两者结合的半自动方法对语料进行标注。自动学习文本结构的方法将在本书的其他部分中讨论。这里,我们仅仅讨论标注的基本知识。一些文本标注仅仅是对基本结构进行标记,例如句子和段落边界,而另外一些方法则要标记更多东西,例如 Penn 树库和 Susanne 语料库对完整的句法结构进行了标注。然而,可以找到的最普遍的语法标记是词性标注,我们将对它给予特别关注。

4.3.1 标注方案

标注文本结构可以采用各种不同的方案。早些时候,由于条件所限,很多方案是在特定应用基础上设计的。比较重要的一个早期例子是 COCOA 格式,用来包含文本的头信息(给出作者、日期、标题等)。这些信息用尖括号(<和>)括起来,第一个字母表示某个域的主要语义。这种格式现在在一些系统中仍然被广泛使用。最常用的语法标注形式是对每个词添加一个词性标记,以此来表示该词的词性,我们接下来就将详细讨论这种标注。这些标记方法一般在每个词后面加一个斜杠或者下划线,之后是一个表示词性的短编码。Penn 树库使用了一种类似于 Lisp 语言的括弧在文本上标注树结构。

然而,到目前为止,最常用也是被推荐最多的标注形式是 SGML(Standard Generalized Markup Language,通用标记语言)。SGML 是一种可以定义文本语法的通用语言,即可以定义它所包含的标记类型。现在到处都可以见到的 HTML 是 SGML 标注的一个实例。文本标注入门(TEI)的主要工作就是定义一种合适的标注方案,从而可以对从诗歌、小说到字典等各种不同体裁的人类文本资源进行标注。XML 是另外一个需要了解的缩写词。XML 是网络程序所用到的 SGML 的一个简单子集。XML 支持商业应用,并且去掉了 SGML 中晦涩和陈旧的内容,降低了复杂性,因此它也完全可以广泛地应用于其他领域。

本书并不打算深入研究 SGML。我们只是给出一些基本知识,以便可以继续讲述原来的问题。SGML 指定每个文档都要有一个文档类型定义(Document Type Definition, DTD),它是文档合法结构的语法。例如,它可以规定一个规则,如一个段落必须包含一个或者多个句子,而不能包含任何其他成分。SGML 分析器会对文档进行验证,以确定是否符合 DTD。但是在统计自然语言处理中,一般会忽略 DTD,通常直接在找到的文本上进行处理。一个 SGML 文档包含一个或者多个元素,这些元素可以递归嵌套。元素一般以一个开始标签开始,以一个结束标签结束,在开始标签和结束标签之间就是文档内容。标签包含在尖括号中,结束标签以一个斜杠开头。开始标签中除了包含标签名之外,还包含额外的属性和属性值信息。下面是一对 SGML 元素的例子:

- a. `<p><s>And then he left.</s>`
`<s>He did not say another word.</s></p>`
- b. `<utt speak="Fred" date="10-Feb-1998">That is an ugly`
`couch.</utt>`

(4.8)

例句(4.8a)中的标签 s 表示句子,标签 p 表示段落,这些都比较常见。例句(4.8b)中有一个拥有属性和属性值的标签。一个元素也可以只包含一个标签(没有配对的结束标签)。在 XML

中,这种空元素必须在标签名后面加一个斜杠来标记。

通常,偶尔使用 SGML 标注的文本时,会希望能解释尖括号中的一些标签而忽略另外一些。必须了解 SGML 的其他一些句法是字符和实体的引用。引用以 & 开头,以分号结束。字符引用通过数字编码来指定一个不在标准 ASCII 字符集(不包括 SGML 保留标注字符)中的字符。实体引用需要一个在 DTD(或者是前面定义过的实体之一)中定义的符号名。实体引用可以扩展到任何文本,但是通常只是通过一个符号名来编码一个特殊字符。例句(4.9)给出了一些字符和实体引用的例子。它们可能出现在一个浏览器中或者打印出来,如例句(4.10)所示。

- a. < is the less than symbol
- b. résumé; (4.9)
- c. This chapter was written on &docdate;.
- a. < is the less than symbol
- b. résumé (4.10)
- c. This chapter was written on January 21, 1998.

关于 SGML 还有许多知识需要了解,4.4 节中给出了一些参考资料,但是我们提供的这方面的知识已经足够应用于本书的内容了。

4.3.2 语法标注

语法分析的第一步是对传统词性类别进行自动语法标注,但是通常对于这样的问题要考虑得更加详尽一些(例如,区别比较级和最高级的形容词,以及单数和复数名词)。本节将着眼于标注集的一些基本性质,解决这样一些问题:已经使用过什么样的标注集?为什么不同的人使用不同的标注集?你应该选择哪一种?

▼ 如何实现自动标注将是第 10 章讲述的主题。

标注集

历史上最有影响的标注集是美国 Brown 语料库所使用的标注集(Brown 标注集)。兰开斯特(Lancaster)大学开发了一系列的标注集,用来对 Lancaster-Oslo-Bergen 语料库进行标注。最近英国国家语料库也使用了他们的标注集(CLAWS1 到 CLAWS5;CLAWS5 又称 c5 标注集)。近来,Penn 树库标注集是计算语言学领域使用最为广泛的标注集。它是 Brown 标注集的一个简化版本。表 4.4 是标注集规模的简明总结。图 4.2 是使用几种不同的标注集对一个例句进行标注的结果。这些标注集都是英语标注集。通常,标注集结合了一种特定语言的形态学特性,所以不能直接应用于其他的语言(然而一些设计思路通常是可以应用的)。许多其他语言的标注集也已经被开发出来了。

表 4.4 各种不同标记集的规模

标注集	基本大小	总标记数
Brown	87	179
Penn	45	
CLAWS1	132	
CLAWS2	166	
CLAWS c5	62	
London-Lund	197	

句子	CLAWS c5	Brown	Penn 树库	ICE
she	PNP	PPS	PRP	PRON(pers,sing)
was	VBD	BEDZ	VBD	AUX(pass,past)
told	VVN	VBN	VBN	V(ditr,edp)
that	CJT	CS	IN	CONJUNC(subord)
the	ATO	AT	DT	ART(def)
journey	NN1	NN	NN	N(com,sing)
might	VM0	MD	MD	AUX(modal,past)
kill	VVI	VE	VB	V(montr,infin)
her	PNP	PPO	PRP	PRON(poss,sing)
.	PUN	.	.	PUNC(per)

图 4.2 按几种不同的标注集对一个例句进行标注

表 4.5 和表 4.6 给出了一些传统的词性标注集,尽管我们不能保证在每个细节上都正确。它们大多是按照字母顺序排列的,但是我们稍微背离了一点字母顺序,以便将其归类。在这些类别里,我们使用其他一些约定,即当一个词不能被划分为更加详细的子类时,它所使用的标记类别要求级别最小。例如,形容词类用来标注形容词,不区分比较级、最高级和单复数。完整的 Brown 标注集由于使用了两个扩大的标注集变得更加庞大。普通标记后面跟一个连字符和一个属性,例如 TL(表示标题词),或者,在外文词的情况下,FW(外文词)标记后面跟一个连字符和一个相关词性标记。另外, Brown 标记方案使用了一个所谓的组合标记(combined tags)来标注图表词,这些图表词可以被看成是具有多重词位的词,例如 *you'll*^①。通常使用加号连起来的两个标记来标注这些术语,对于否定形式则要在标记后面附加一个 * 号。所以 *isn't* 被标注为 BEZ *, *she'll* 被标注为 PPS + MD。另外, *children's* 之类的所有格形式使用以 \$ 结束的标记来标注。通常,这些标记明显地继承了一个基本的非所有格标记,例如上面这个例子(*children's*)中就是 NNS \$。这些扩展标注集的技术在比较中都被忽略了。

① 可以和上面的讨论进行对比。这种方法也应用于其他语料库,比如 London-Lund 语料库,但是最近有把图表词切分开的趋势。

表 4.5 不同标注集的比较:形容词、副词、连词、限定词、名词和代词的标注符号

类别	例子	Claws c5	Brown	Penn
Adjective	happy, bad	AJO	JJ	JJ
Adjective, ordinal number	sixth, 72nd, last	ORD	OD	JJ
Adjective, comparative	happier, worse	AJC	JJR	JJR
Adjective, superlative	happiest, worst	AJS	JJT	JJS
Adjective, superlative, semantically	chief, top	AJO	JJS	JJ
Adjective, cardinal number	3, fifteen	CRD	CD	CD
Adjective, cardinal number, one	one	PNI	CD	CD
Adverb	often, particularly	AVO	RB	RB
Adverb, negative	not, n't	XX0	*	RB
Adverb, comparative	faster	AVO	RBR	RBR
Adverb, superlative	fastest	AVO	RBT	RBS
Adverb, particle	up, off, out	AVP	RP	RP
Adverb, question	when, how, why	AVQ	WRB	WRB
Adverb, degree & question	how, however	AVQ	WQL	WRB
Adverb, degree	very, so, too	AVO	QL	RB
Adverb, degree, postposed	enough, indeed	AVO	QLP	RB
Adverb, nominal	here, there, now	AVO	RN	RB
Conjunction, coordination	and, or	CJC	CC	CC
Conjunction, subordinating	although, when	CJS	CS	IN
Conjunction, complementizer <i>that</i>	that	CJT	CS	IN
Determiner	this, each, another	DT0	DT	DT
Determiner, pronoun	any, some	DT0	DTI	DT
Determiner, pronoun, plural	these, those	DT0	DTS	DT
Determiner, prequalifier	quite	DT0	ABL	PDT
Determiner, prequantifier	all, half	DT0	ABN	PDT
Determiner, pronoun or double conj.	both	DT0	ABX	DT (CC)
Determiner, pronoun or double conj.	either, neither	DT0	DTX	DT (CC)
Determiner, article	the, a, an	ATO	AT	DT
Determiner, postdeterminer	many, same	DT0	AP	JJ
Determiner, possessive	their, your	DPS	PP\$	PRP\$
Determiner, possessive, second	mine, yours	DPS	PP\$\$	PRP
Determiner, question	which, whatever	DTQ	WDT	WDT
Determiner, possessive & question	whose	DTQ	WP\$	WP\$
Noun	aircraft, data	NN0	NN	NN
Noun, singular	woman, book	NN1	NN	NN
Noun, plural	women, books	NN2	NNS	NNS
Noun, proper, singular	London, Michael	NP0	NP	NNP
Noun, proper, plural	Australians, Methodists	NP0	NPS	NNPS
Noun, adverbial	tomorrow, home	NN0	NR	NN
Noun, adverbial, plural	Sundays, weekdays	NN2	NRS	NNS
Pronoun, nominal (indefinite)	none, everything, one	PNI	PN	NN
Pronoun, personal, subject	you, we	PNP	PPSS	PRP
Pronoun, personal, subject, 3SG	she, he, it	PNP	PPS	PRP
Pronoun, personal, object	you, them, me	PNP	PPO	PRP
Pronoun, reflexive	herself, myself	PNX	PPL	PRP
Pronoun, reflexive, plural	themselves, ourselves	PNX	PPLS	PRP
Pronoun, question, subject	who, whoever	PNQ	WPS	WP
Pronoun, question, object	who, whoever	PNQ	WPO	WP
Pronoun, existential there	there	EX0	EX	EX

粗略地看一眼就会发现,不同标注集之间的区别很明显。部分原因是标注集的整体规模。一个大的标注集明显地可以进行更加细粒度的区分,但这并不是惟一的区分。在不同的领域可能选择不同的标注集来进行区分。例如,c5 标注集在规模上比 Penn 树库标注集大,在某些领域它区分得很细,但是在另外一些领域它就区分得不那么细了。例如,Penn 标注集区分 9 种标点标记,而 c5 仅仅区分 4 种。这也表明了不同标注集标注的侧重点的不同。标注集的根本差别在于如何对某些词进行分类。例如,Penn 标注集简单地把从句连词当成介词(和生成语言学一致),c5

标注集则把它们保留为单独一类。不仅如此,还把它们和其他种类的连词一起进行隐含分组。这里提到的隐含分组的概念是指所有的标注集都非正式地通过同样的一个或者一对字母开头,以表明某些标注集之间的关系。这种分组是隐含的,尽管看起来很明显,但它们只是形式上区别符号标记,程序通常并不使用它们。然而,在另外的一些标注集里,例如英语国际语料库(Greenbaum 1993)所使用的标注集,采用了一种清楚地带有特征表达式属性的高水平标记系统。早期的标注集在许多领域都做了非常仔细的区分,例如各种修饰词和限定词,这样的划分有时只和很少的一些词有关系。最近的标注集在这些方面逐渐减少了这种区分。

表 4.6 不同标注集的比较:动词、介词、标点和符号的标注符号。‘not’条目表示在标注中忽略这项,或者是不作为一个单独的词次进行标注

类别	例子	Claws c5	Brown	Penn
Verb, base present form (not infinitive)	take, live	VVB	VB	VBP
Verb, infinitive	take, live	VVI	VB	VB
Verb, past tense	took, lived	VVD	VBD	VBD
Verb, present participle	taking, living	VVG	VBG	VBG
Verb, past/passive participle	taken, lived	VVN	VBN	VBN
Verb, present 3SG -s form	takes, lives	VVZ	VBZ	VBZ
Verb, auxiliary do, base	do	VDB	DO	VBP
Verb, auxiliary do, infinitive	do	VDB	DO	VB
Verb, auxiliary do, past	did	VDD	DOD	VBD
Verb, auxiliary do, present part.	doing	VDG	VBG	VBG
Verb, auxiliary do, past part.	done	VDN	VBN	VBN
Verb, auxiliary do, present 3SG	does	VDZ	DOZ	VBZ
Verb, auxiliary have, base	have	VHB	HV	VBP
Verb, auxiliary have, infinitive	have	VHI	HV	VB
Verb, auxiliary have, past	had	VHD	HVD	VBD
Verb, auxiliary have, present part.	having	VHG	HVG	VBG
Verb, auxiliary have, past part.	had	VHN	HVN	VBN
Verb, auxiliary have, present 3SG	has	VHZ	HVZ	VBZ
Verb, auxiliary be, infinitive	be	VBI	BE	VB
Verb, auxiliary be, past	were	VBD	BED	VBD
Verb, auxiliary be, past, 3SG	was	VBD	BEDZ	VBD
Verb, auxiliary be, present part.	being	VBG	BEG	VBG
Verb, auxiliary be, past part.	been	VBN	BEN	VBN
Verb, auxiliary be, present, 3SG	is, 's	VBZ	BEZ	VBZ
Verb, auxiliary be, present, 1SG	am, 'm	VBB	BEM	VBP
Verb, auxiliary be, present	are, 're	VBB	BER	VBP
Verb, modal	can, could, 'll	VM0	MD	MD
Infinitive marker	to	TO0	TO	TO
Preposition, to	to	PRP	IN	TO
Preposition, for, above	for, above	PRP	IN	IN
Preposition, of	of	PRF	IN	IN
Possessive	's, '	POS	\$	POS
Interjection (or other isolate)	oh, yes, mmm	ITJ	UH	UH
Punctuation, sentence ender	. ! ?	PUN	.	.
Punctuation, semicolon	;	PUN	.	.
Punctuation, colon or ellipsis	: ...	PUN	:	:
Punctuation, comma	,	PUN	,	,
Punctuation, dash	-	PUN	-	-
Punctuation, dollar sign	\$	PUN	not	\$
Punctuation, left bracket	({ [	PUL	(	(
Punctuation, right bracket	)] }	PUR	)	)
Punctuation, quotation mark, left	“	PUQ	not	“
Punctuation, quotation mark, right	”	PUQ	not	”
Foreign words (not in English lexicon)		UNC	(FW-)	FW
Symbol	[fj] *		not	SYM
Symbol, alphabetical	A, B, c, d	ZZ0		
Symbol, list item	A A. First			LS

标注集的设计

应该使用什么特征来指导设计标注集呢?通常,一个标注集同时依据分类目标特征和预测特征来进行编码。分类目标特征告诉用户关于一个词的语法类别的有用信息,预测特征是对预测上下文中其他词语特性有用的特征。它们通常是相互交叉的,但是没有必要等同看待。

词性的概念实际上很复杂,因为这个概念的引出可以有不同的背景,比如语义背景、句法分布背景或者词形学背景。通常词性的概念是相互冲突的。出于预测的目的,我们希望使用可以最佳预测附近词语特性的词性定义,这样它就是严格可推测的分布标记。但是,实际上人们经常使用可以反映概念或者词形学标准的标记。例如英语中以-ing结尾的现在分词可以作为动名词,它的行为和名词一样。Brown语料库中通常将其标记为VBG,但是似乎它作为一个动词的分词用法更加合适。这种情况甚至出现在下面这些明显的复合名词中:

Fulton/NP-TL County/NN-TL Purchasing/VBG Department/NN (4.11)

在理想情况下,我们希望给特殊分布的词汇标上特殊的标记,这样在其他处理中就可以使用特殊标记的信息。也就是说,一些标记太过粗略以致不能很好地进行预测,例如在Penn树库标注集中的标记。例如,补语化成分的引导词*that*和一般介词有不一样的分布,它有非常特殊的作用。程度副词和否定词*not*和一般的副词也有不一样的分布,它们也拥有非常不同的作用。但是,在上述标注集中并没有对这些进行区分。人们经常根据直觉添加或者去除特殊词性来更改标注集。例如,(Charniak 1996)质疑Penn树库为什么对助动词使用和其他动词一样的标记,因为已知助动词有非常特殊的作用,因此采用AUX标签来重新标注助动词。通常,在词性系统中标注集进行特别的改动是否可以增大预测值,并没有系统的计算方法可以计算。只要使用同样的标注集进行预测和分类,做这样的改变就有利也有弊,好比一把双刃剑:分裂标记可以得到更详细的划分,为预测提供更多有用信息,但是同时这也使得分类工作变得更不容易^①。因此,在标注集规模和自动标注的效果之间没有一个简单的关系。

4.4 深入阅读

Brown语料库(Brown大学当代美国英语标准语料库)由从1961年以来的100多万美国书面英语单词组成。它是由W. Nelson Francis和Henry Kučera进行文档处理和编辑的,参见(Francis and Kučera 1964; Kučera and Francis 1967; Francis and Kučera 1982)。早期处理Brown语料库细节的阐述是来自于Henry Kučera的一封电子邮件(1996年9月26日被Katsuhide Sonoda发到了语料库邮件列表里)。20世纪70年代建立的LOB(Lancaster-Oslo-Bergen)语料库相当于Brown语料库的英国英语的复制品,参见(Johansson et al. 1978; Garside et al. 1987)。

在信息提取中识别正确的名字是一个主要的主题。这方面的介绍参见(Cardie 1997)。

Susanne语料库所使用的词次处理规则集是一个仔细设计并经过实验检验的集合,参见(Sampson 1995:52~59)。

Nunberg(1990)提出了一个关于标点符号重要性的语言学观点。在(Crowley et al. 1995:7~9)

^① 除非一个类别可以聚成两个完全不同分布的子类。在这种情况下,把这个类别划分为两个子类别有时会简化分类问题。

中可以找到什么是语言学上的词的介绍性讨论。Lyons(1968:194~206)提供了一个更加深入的讨论。关于连字符的小节中提到的例子大部分都是 Dow Jones 新闻专线的真实例子。其他的是由 Robert Amsler 和 Mitch Marcus 于 1996 年从发到语料库列表的电子邮件里得到的,我们在这里表示感谢。

我们可以获得许多现有的分析词形学系统,一些已经列在了网站上。知识贫乏条件下进行词干化的高效率方法可以在 Kay and Röscheisen(1993)中找到。Sproat(1992)包含了自然语言处理语法问题的讨论,同时也是前面的德语复合结构例子的来源。

COCOA(COunt and COncordance on Atlas)格式在源于 ICAME 的语料库中和相关的软件,例如 LEXA 中被使用,参见(Hickey 1993)。

许多参考资料中都有对 SGML 和 XML 的介绍,如(Herwijnen 1994; McGrath 1997; St. Laurent 1998)。文本标注入门(TEI)(1994 年 P3 版)由 McQueen and Burnard(1994)发表,并且在书中第 2 章包含了对 SGML 的非常易读的介绍。然而,与其阅读实际的标注指南,不如阅读教程,例如(Ide and Véronis 1995),或者从网站上查阅资料。TEI 标准完全版的复杂性超过了现有的几乎所有标准。近来的发展包括试图把原有标准削减到一个人类可用版本的 TEILite,还有语料库标注标准,一个专门为语言工程语料库设计的与 TEI 一致的 SGML 例子。

(Garside et al. 1987)描述了 CLAWS(Constituent-Likelihood Automatic Word-tagging System)的早期工作和它的标注集。上面提到过的比较新的 c5 标注集是从(Garside 1995)中选取的。Brown 标注集可以在(Francis and Kučera 1982)中找到,Penn 标注集可以在(Marcus et al. 1993)中找到,更加详细的内容在(Santorini 1990)中。

本书并不是一本介绍如何使用语料库来进行语言学研究的读物(尽管它包含很多对这类研究非常有用的方法和算法)。近来已经出现了一批语料库语言学方面的新书,如(McEnery and Wilson 1996; Stubbs 1996; Biber et al. 1998; Kennedy 1998; Barnbrook 1996)。这些书中也包含了许多语料库设计方面的主题,例如我们提到过的样本选取和平衡问题。如果想找一篇专门针对设计一个有代表性语料库的文章,可以阅读(Biber 1993)。

(Garside et al. 1987)的附录 B 中和 AMALGAM 工程网页(见本书配套网站)中收集了许多不同标注集的细节。AMALGAM 网站也包含对其所用的词次规则的介绍,我们可以将其看成一个启发式的句子分割和词次识别的例子。(Grefenstette and Tapanainen 1994)讨论了另一种词次标记方法,并且显示了采用简单的知识贫乏启发式方法的实验结果。

4.5 习题

- 4.1 [★★]文中提到过,为了大多数目的的需要,我们把一些连字符连接起来的内容当成词(例如, *co-worker*, *Asian-American*),另外一些则不作为词处理(例如, *ain't-it-great-to-be-a-Texan*, *child-as-required-yuppie-possession*)。如果要在一个语料库中找出由连字符连接起来的形式,并且提出一些基本原则,判断哪些应该当成词,而哪些不应该当成词,你的判定原则的根据是什么? 不同的选择可能适合不同的需求。找出一些方法来确定什么样的连接序列应该被打散,例如,复合名词的非结尾元素:

[*N* [*child-as-required-yuppie-possession*] *syndrome*]

- 4.2 [★★ 适合语言学家]找一些你感兴趣的语言学问题(non-constituent coordination, ellip-

sis, idioms, heavy NP shift, pied-piping, verb class alternations, 等等)。可以期望在一个通用的语料库中找到和这些问题相关的有用数据吗? 为什么能或者为什么不能? 如果你认为这是可能的, 那么是否有合理的方法可以在生语料或者有语法结构的语料中搜索到上述语言现象的例子? 如果这两个问题的答案都是肯定的, 那么在一个语料库中寻找例子并且把找到的任何有趣的东西写成报告。

- 4.3 [★★]设计一个句子边界检测算法, 评价它是否成功。在 ACL-DCI CD-ROM (Church and Liberman 1991) 的 *Wall Street Journal* 一节的构造中, 使用了一个相当简单的句子边界检测算法, 结果没有经过手工改正, 还存在很多错误。如果你能得到这个语料库, 可以用你的结果和语料库中标好的句子边界进行比较。如果比较幸运, 你可能会做出一个效果更好的系统!

第二部分 词 法

第 5 章 搭配

第 6 章 统计推理:稀疏数据集上的 n 元语法模型

第 7 章 语义消歧

第 8 章 词汇获取

第5章 搭 配

“搭配”是由两个或两个以上的词所组成的语言表示,相当于说某些事情的习惯方式。或者,用 Firth (1957: 181) 的话来说:“一个特定词的搭配就是对这个词的惯用位置的描述”。搭配包括:名词短语,如 *strong tea* 和 *weapons of mass destruction*, 动词短语,如 *to make up*, 还有一些固定短语,如 *the rich and powerful*。特别有趣的是,一些词汇使用的模式很微妙,不容易解释,但是母语说话者能够区分,例如为什么用 *a stiff breeze* 而不用 *a stiff wind* (而 *a strong breeze* 和 *a strong wind* 都是对的), 或者为什么用 *broad daylight* (而不用 *bright daylight* 或 *narrow darkness*)。

搭配由有限的复合构词法 (compositionality) 所描述。在自然语言的表述中,如果可以从各组成部分的意思推测出整体表述的意思,那么我们称这个语言表述是复合构成的。搭配不完全是复合构成的,因为它通常还有别的含义。在例子 *strong tea* 中, *strong* 含有了 *rich in some active agent* 的含义,这和它的基本含义 *having great physical strength* 稍微有些区别。成语或俚语是非复合构词法最极端的例子。像 *to kick the bucket* 或 *to hear it through the grapevine* 这样的成语,仅仅和它们表面的意思有一些间接的历史关系。当使用这些成语时,我们不会照 *bucket* 和 *grapevine* 的字面意思来理解它们。大部分的搭配显示了比较温和的非复合构词法的结构,比如在本书前面使用的 *international best practice* 的例子。它的意思非常接近每部分的组合意思,但是仍然有别的含义。通常它是指管理效率,而不是用来描述如烹饪技术之类的内容,虽然这些意思和字面意思也是一致的。

在搭配 (collocation) 和术语 (term)、技术术语 (technical term)、专有名词短语 (terminological phrase) 的概念之间有很多交叉的地方。就像它们的名字所指出的一样,后三项通常是在技术领域中抽取到的搭配 (这个过程又称术语识别)。读者应该注意到,在信息检索中 *term* 有不同的意思,这里是指词或者短语。我们在本章中将要用到的只是它的狭义说法。

搭配对很多应用都有重要的作用。例如,自然语言生成 (natural language generation), 保证输出看起来是自然的,并且可以避免像 *powerful tea* 或 *to take a decision* 这样的错误; 计算词典编撰学 (computational lexicography), 自动识别词典条目中重要的搭配; 句法分析 (parsing), 含有自然搭配的分析结果可以被优先选择; 语料库语言学的研究 (corpus linguistic research), 通过语言对一些社会现象进行研究, 参见 (Stubbs 1996)。

我们之所以对搭配产生这么大的兴趣,部分原因在于,这是一个在 Saussure 和 Chomsky 开创的结构语言学传统中被忽视的领域。然而,在英国语言学的传统中,相应的研究者 Firth, Halliday 和 Sinclair 在搭配现象的研究上花费了大量的精力。结构语言学专注于对词组和句子属性的一般化抽象。与此相反, Firth 的上下文理论 (Contextual Theory of Meaning) 强调了上下文的重要性,包括社会背景的上下文 (对应于理想的说话者), 口语和文本篇章的

上下文(对应于孤立的句子),还有对搭配发现很重要的上下文的环境词(Firth著名的格言是“一个词是被它周围伴随的词所刻画的”)。这些上下文特性很容易被结构语言学家典型的抽象操作忽略掉。

Halliday 的关于 *strong tea* 和 *powerful tea* 的例子很好地说明了这个问题,参见(Halliday 1966:150),这是语言学上下文观点中一个很重要的例子。英语中习惯说 *strong tea*,而不说 *powerful tea*,任何一个说英语的人都能理解后面一种是非习惯性的表达。可以看出,我们不能从这两种表达方法中抽取出我们感兴趣的英语的结构属性。但是,这两种表达方法的对比可以告诉我们一些有趣的东西:关于我们的文化中对不同类型内容的本质看法(为什么使用 *powerful* 修饰毒品,如海洛因,而不用它来修饰烟草、茶和咖啡?)。并且,对于那些希望学习纯正的英语的学生来说,把这些表达方式之间的对比教给他们是非常重要的。遵循 Firth 方法的英国语言学家们感兴趣的正是这类问题:语言使用和语言教学中的社会背景。

在本章中,我们将会介绍一些识别搭配对的方法:使用频率信息的搭配识别,基于含义和主词搭配词之间距离的识别,基于假设测试和互信息的识别。然后,我们将会回到搭配是什么这个问题上来,在更深的层次讨论不同的定义,并测试一个短语是否可以被认为是一个搭配。本章结尾列出了相关内容的深入阅读材料,即一些我们在书中没有涵盖的相关文献。

本章所举的例子中使用的参考语料库是由纽约时报新闻专线的语料组成的,该语料库包含 1990 年 8 月到 1990 年 11 月的内容。这个语料库大约有 115 MB 的文本,包含大约 1400 万个词。每种方法都会应用到这个语料库上,可以方便地比较结果。对于本章的大部分内容,所举的纽约时报例子只是从固定的两词短语(bigram)中抽取搭配。我们这样做仅仅是因为方便,牢记这一点对理解这些例子很重要。通常,固定的和不定的词语组合都可以是搭配,而且均值和方差与本章的联系并不大。

5.1 频率

在一个文本语料库中寻找搭配的最简单的方法就是计数。如果两个词在一起出现了很多次,那么这就是一个证据,说明它们有特殊的功能,这种功能不能简单解释为两个词合并而导致的结果。

可以预计的是,仅仅选择最频繁出现的二元组并不理想,表 5.1 中就给出了这样的例子。表中显示了在语料库中最频繁出现的二元组(两个连续词的序列)和它们的出现频率。除了 *New York* 以外,所有的二元组都是一对功能词。

有一个非常简单的启发式规则可以大大提高结果的准确率。(Justeson and Katz 1995b)中通过一个词性过滤器来过滤候选短语,这个过滤器只允许可能是“短语”^①的模式通过。Justeson and Katz(1995b:17)提出了表 5.2 中的模式。在每个模式后面都有一个例子,这个例子源于所用的测试文本。在这些模式中,A 代表形容词,P 代表前置词,N 代表名词。

^① 类似的想法可以在(Ross and Tukey 1975)和(Kupiec et al. 1995)中找到。

表 5.1 单纯依赖于频率的搭配发现。 $C(\cdot)$ 表示语料库中统计对象出现的频率

$C(W^1 W^2)$	W^1	W^2
80 871	of	the
58 841	in	the
26 430	to	the
21 842	on	the
21 839	for	the
18 568	and	the
16 121	that	the
15 630	at	the
15 494	to	be
13 899	in	a
13 689	of	a
13 361	by	the
13 183	with	the
12 622	from	the
11 428	New	York
10 007	he	said
9775	as	a
9231	is	a
8753	has	been
8573	for	a

表 5.2 搭配过滤器的词性标记模式。Justeson 和 Katz 使用这些模式在频繁出现的词串中间识别可能的搭配

标记模式	示例
A N	<i>linear function</i>
N N	<i>regression coefficients</i>
A A N	<i>Gaussian random variable</i>
A N N	<i>cumulative distribution function</i>
N A N	<i>mean squared error</i>
N N N	<i>class probability function</i>
N P N	<i>degrees of freedom</i>

表 5.3 显示了经过过滤器过滤之后出现频率排名最高的短语。这个结果好得令人惊异。表中只有 3 个二元组是复合短语: *last year*, *last week*, *first time*。*York City* 是使用 Justeson 和 Katz 的方法过滤的一个特殊产物。完整的方法将会搜寻满足一个词性标记模式的最长序列, 因此我们可以找到更长的短语 *New York City*, 它包含了 *York City*。

包含 *strong* 和 *powerful* 的前 20 个排名最高的短语都是 AN(在这里 A 表示 *strong* 或者 *powerful*)形式的。我们已经把这个结果列在表 5.4 中。

表 5.3 搭配发现: Justeson 和 Katz 的词性过滤器

$C(W^1 W^2)$	W^1	W^2	标记模式
11 487	New	York	A N
7261	United	States	A N
5412	Los	Angeles	N N
3301	last	year	A N
3191	Saudi	Arabia	N N
2699	last	week	A N
2514	vice	president	A N
2378	Persian	Gulf	A N
2161	San	Francisco	N N
2106	President	Bush	N N
2001	Middle	East	A N
1942	Saddam	Hussein	N N
1867	Soviet	Union	A N
1850	White	House	A N
1633	United	Nations	A N
1337	York	City	N N
1328	oil	prices	N N
1210	next	year	A N
1074	chief	executive	A N
1073	real	estate	A N

表 5.4 模式 'strong w' 和 'powerful w' 中最常出现的名词 w

w	$C(\text{strong}, w)$	w	$C(\text{powerful}, w)$
support	50	force	13
safety	22	computers	10
sales	21	position	8
opposition	19	men	8
showing	18	computer	8
sense	18	man	7
message	15	symbol	6
defense	14	military	6
gains	13	machines	6
evidence	13	country	6
criticism	13	weapons	5
possibility	11	post	5
feelings	11	people	5
demand	11	nation	5
challenges	11	forces	5
challenge	11	chip	5
case	11	Germany	5
supporter	10	senators	4
signal	9	neighbor	4
man	9	magnet	4

本节表述的基于均值和方差的方法着眼于一种特殊的模式,在这个模式中两个词之间的距离是可以变化的。如果这样的距离模型是可行的,那么我们就可以说,像 *knock...door* 这样的搭配不必一定认为是固定的短语。在本章的最后,我们将会再讨论这个问题,并且更深入地讨论什么是搭配。

发现 *knocked* 和 *door* 之间的关系的一种方法是,计算语料库中两个词之间的偏移量(有符号的距离)的均值和方差。均值就是简单的平均偏移量。在例句(5.1)中,计算在 *knocked* 和 *door* 之间的平均偏移量:

$$\frac{1}{4}(3 + 3 + 5 + 5) = 4.0$$

我们假设 *Donaldson's* 是 3 个词组成的: *Donaldson*, 所有格符号和 *s*, 事实上我们就是这样计算的。如果有一个 *door* 出现在 *knocked* 前面,那么偏移量会被认为是一个负值。例如, *the door that she knocked on* 的值是 -3。我们这样把分析限制在中心词 *knocked* 周围的 9 个词的窗口上。

方差衡量的是单独的偏移量偏离均值的距离。下面的公式可用来估计方差:

$$s^2 = \frac{\sum_{i=1}^n (d_i - \bar{d})^2}{n - 1} \quad (5.2)$$

公式中 n 是这两个词的同现次数, d_i 是同现 i 的偏移量, $\bar{d}$ 是样本偏移量的均值。如果在所有情况下样本的偏移量是相同的,那么方差为零。如果偏移量是随机分布的(在这种情况下,两个词偶然同现,而不是由于特殊的关系),那么方差将会较大。依照惯例,我们使用样本偏差 $s = \sqrt{s^2}$, 它是方差的平方根,用来评价两个词之间的偏移量是怎样随机变化的。对于上面的情况,4 个 *knocked/door* 的例子的偏差是 1.15:

$$s = \sqrt{\frac{1}{3}((3 - 4.0)^2 + (3 - 4.0)^2 + (5 - 4.0)^2 + (5 - 4.0)^2)} \approx 1.15$$

均值和偏差特性化了语料库中两个词之间距离的分布。我们可以使用这个信息来发现搭配,具体的方法是通过寻找带有低偏差值的词对。一个低的偏差值意味着这两个词通常会以大致相同的距离出现。零偏差意味着这两个词总是以相同的距离出现。

我们也可以这样解释方差信息:方差是关于一个词相对于其他词分布峰值情况的量度。图 5.2 显示了我们感兴趣的三种情况。*strong* 关于 *opposition* 的分布在位置 -1 有一个峰值(对应的短语是 *strong opposition*)。因此 *strong* 关于 *opposition* 的方差比较小($s = 0.67$)。均值为 -1.15 表明 *strong* 通常在 -1 位置出现(忽略在 -4 位置出现一次带来的噪声)。

我们考虑把感兴趣的词限制在中心词周围 9 个词的范围内。这是因为搭配在本质上讲还是一个局部的现象。还要注意的,当我们考虑两个不同词汇之间的关系时,在 0 位置总是得到 0 计数。举前例来说,这是因为在上下文中 *strong* 不能出现在 0 位置上,这个位置已经被 *opposition* 占用了。

接下来关注图 5.2 中的第二个二元组。*strong* 相对于 *support* 的分布被画出来,在几个负位置具有比较大的值。例如,在 -2 位置有大约 20 的统计值,这是由于使用 *strong leftist support* 和 *strong business support* 之类的短语引起的。由于它的较大的灵活度,我们得到了一个较高的偏差 $s(1.07)$ 和一个在位置 -1 和 -2 之间的均值(-1.45)。

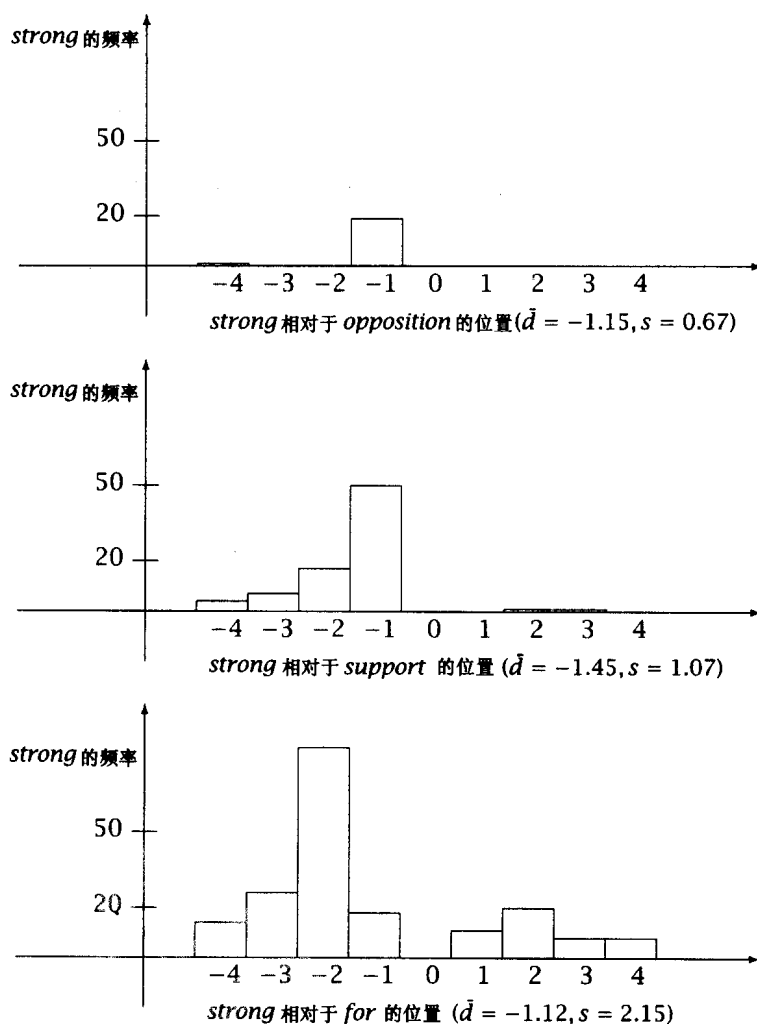


图 5.2 strong 相对于 3 个词的位置的直方图

最后, *strong* 相对于 *for* 的情况出现了更加均匀分布。在这里有一个趋势, *strong* 出现在 *for* 的前面(因此它有一个负的均值, -1.12), 但是它可以出现在 *for* 周围的任何地方。一个高的偏差值 $s = 2.15$ 表明了它的这种灵活性。这样就表明 *for* 和 *strong* 不能形成我们感兴趣的搭配。

表 5.5 中的词对表明了通过上述方法可以找到的一些搭配的类型。如果均值接近 1.0 并且偏差较低, 就像 *New York* 这样的例子, 那么这种短语类型也可以用 Justeson 和 Katz 的基于频率的方法来发现。如果均值远远大于 1.0, 那么一个低偏差预示了我们感兴趣的短语。例如, 词对 *previous/games* (距离是 2) 对应的短语是 *in the previous 10 games* 和 *in the previous 15 games*; 词对 *minus/point* 对应的短语是 *minus 2 percentage points*, *minus 3 percentage points* 等; 词对 *hundreds/dollars* 对应的短语是 *hundreds of billions of dollars* 和 *hundreds of millions of dollars*。

表 5.5 基于均值和方差方法的搭配发现。12 个词对距离的样本的方差和均值

s	d	计数	第一个单词	第二个单词
0.43	0.97	11 657	New	York
0.48	1.83	24	previous	games
0.15	2.98	46	minus	points
0.49	3.87	131	hundreds	dollars
4.03	0.44	36	editorial	Atlanta
4.03	0.00	78	ring	New
3.96	0.19	119	point	hundredth
3.96	0.29	106	subscribers	by
1.07	1.45	80	strong	support
1.13	2.57	7	powerful	organizations
1.01	2.00	112	Richard	Nixon
1.05	0.00	10	Garrison	said

高偏差预示着,在词对的两个词之间并不存在我们感兴趣的联系,就像表 5.5 中的 4 个高偏差的例子那样。注意,在这里均值趋向于零,符合一个均匀分布的期望值。处于中间情况的例子更加有趣,这样的词对在搭配分布的几个距离上有大的统计值。我们已经在图 5.2 中看到了 *strong* { *business* } *support* 的例子。这样的情况也在其他 3 个中等方差的例子里出现,它们是 *powerful* { *lobbying* } *organizations*, *Richard* { *M.* } *Nixon* 和 *Garrison said/said Garrison* (回忆一下,我们把符号 *Richard M. Nixon* 作为 4 个词次: *Richard*, *M.*, *Nixon*)。

本章中介绍的这种基于方差的搭配发现方法应该归功于 Smadja。有些内容我们已经稍微做了简化。特别是 Smadja(1993)使用了一个额外的限制,过滤了在一些位置上“肥胖”的峰值。更确切地说,是那些周围位置上没有较低峰值的柱状图(例如,图 5.2 中对于 *strong/for* 在 -2 位置上的柱状图)。Smadja(1993)成功地把该方法应用在术语抽取(估计有 80% 的准确率)和在自然语言生成中确定合适的短语,参见(Smadja and McKeown 1990)。

Smadja 的搭配的概念没有其他一些研究者的严格。对于组合 *knocked/door* 来说,它可能不是一个我们想要的严格术语意义上的搭配,但是对于文本生成的目的来说它可能非常有用。如果我们想要寻找这样的词汇组合,在这些词汇组合之间的联系比固定短语之间的联系更加松散,并且在它们之间插入的符号及其相对位置都是可变的,那么这种基于方差的搭配发现方法是一个正确的选择。

5.3 假设检验

我们掩盖至今的一个难题是,高频率和低方差可能是偶然出现的。如果一个频繁出现的二元组的两个构成词也是频繁出现的词,例如 *new companies* (*new* 和 *companies* 都频繁出现),那么我们可以预期这两个词同现很多次仅仅是一个偶然现象,它们甚至不能形成一个搭配。

事实上,我们真正想知道的是两个词同现的次数是否比偶然同现的次数更多。评价一个事件是否是偶然事件是统计学的经典问题之一。我们通常用假设检验来评价。首先,我们定义一个零假设(null hypothesis) H_0 ,假设这两个词的同现与它们的偶然同现没有关系。在这个假设之下,我们计算事件出现的概率 p 。如果 p 非常小(典型情况是,如果在 $p < 0.05, 0.01$,

0.005或0.001的置信水平(significance level)之下)就否定 H_0 ,在其他情况下保留 H_0 ①。

我们一定要注意,在同一个时间考察两个事物是一种数据分析的模式。正如以前我们在数据中寻找一个特殊的模型时,也要考虑数据量的多少。即使我们有了一个非常好的模型,如果没有足够的数据来确定得到的参数不是由偶然性引起的,那么也要对这个模型打一个折扣。

怎样才能把假设检验的方法应用到寻找搭配的问题上呢?首先,我们定义一个零假设,它表明了如果两个词不能形成一个搭配会是什么情况。对于这样的两个词的自由组合,我们假设词 w_1 和词 w_2 分别都是完全独立产生的。那么,它们偶然在一起的概率可以由以下的公式给出:

$$P(w^1 w^2) = P(w^1)P(w^2)$$

这个模型隐含着同时出现的概率仅仅是单独的词出现的概率的乘积。在本节最后的讨论中会看到,这个模型是一个非常简化的模型,并且看上去不精确,但是我们现在要把它作为我们的零假设。

5.3.1 t 检验

下面我们需要一个统计检测来估计一个确定的并列关系出现的可能性,或者它不出现的可能性。对于搭配发现,一个已经广泛使用的测试就是 t 检验。 t 检验着眼于样本的均值和方差,零假设规定样本服从均值为 μ 的分布。 t 检验考虑了在期望均值和观测均值之间的不同,使用数据的方差来衡量,并且可以得出,在假设样本服从均值为 μ 的正态分布的情况下,得到具有这样一个均值和方差的样本(或者得到一个具有非常极端的均值和方差的样本)的可能性有多大。为了确定得到我们的样本(或者是一个很极端的样本)的概率,计算统计量 t :

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}} \quad (5.3)$$

在这里, $\bar{x}$ 是样本均值, s^2 是样本方差, N 是样本的大小, μ 是分布的均值。如果统计量 t 足够大,我们可以拒绝零假设。通过查表,可以准确地得出 t 的大小,我们已经把 t 分布编辑到了附录 A 中(或者可以使用统计学参考书中的更加专业的表,也可以使用合适的计算软件进行计算)。

这里有一个使用 t 检验的例子。我们的零假设是,男性人口的平均身高是 158 cm。有一个 200 人的样本, $\bar{x} = 169$ 并且 $s^2 = 2600$,我们希望知道这个样本是否从一般的人群(符合零假设)中得来,或者它是从一个不同的较小的人群中得到。对应上面的公式,可得统计值 t :

$$t = \frac{169 - 158}{\sqrt{\frac{2600}{200}}} \approx 3.05$$

如果在置信度 $\alpha = 0.005$ 的水平查表,会得到 t 的值为 2.576②。因为我们得到的 t 值大于 2.576,所以可以有 99.5% 的信任度拒绝零假设。因此可以说,这个样本不是由均值为 158 cm

① 0.05 的置信水平通常是科学实验中最弱的证据。对于统计自然语言处理的任务来说,大规模的数据应用意味着我们可以经常期望得到一个更高级别的置信水平。

② 均值为 200,自由度为 199 的样本,它近似的和自由度无穷的样本有相同的 t 值。据此查表得 2.576。

的人群中得到的,并且我们判断错误的概率小于0.5%。

为了显示怎样把 t 检验应用到搭配识别上,我们计算了 *new companies* 的 t 值。那么要测量均值和方差的样本是什么呢? 这里有一个标准方法,可以把 t 检验扩展到使用比例和计数的方法上。我们把一个文本语料库看成 N 个二元组的长序列,那么样本就是一个随机变量的取值指示器,当我们感兴趣的二元组出现的时候它的值为1,在其他情况下为0。

使用最大似然估计的方法,我们可以计算出 *new* 和 *companies* 的概率如下。在我们的语料库中, *new* 出现 15 828 次, *companies* 出现 4675 次,并且总共有 14 307 668 个词次(token)。

$$P(\text{new}) = \frac{15\,828}{14\,307\,668}$$

$$P(\text{companies}) = \frac{4675}{14\,307\,668}$$

零假设是, *new* 和 *companies* 是独立出现的。

$$\begin{aligned} H_0: P(\text{new companies}) &= P(\text{new})P(\text{companies}) \\ &= \frac{15\,828}{14\,307\,668} \times \frac{4675}{14\,307\,668} \approx 3.615 \times 10^{-7} \end{aligned}$$

如果零假设为真,那么随机产生二元词组,指定 *new companies* 的值为1,其他情况下的值为0的过程,就是一个有效的 Bernoulli 判据。对于 *new company*, 出现概率 $p = 3.615 \times 10^{-7}$ 。这个分布的均值是 $\mu = 3.615 \times 10^{-7}$, 并且它的方差是 $\sigma^2 = p(1-p)$ (参见 2.1.9 节), 近似等于 p 。近似表达 $\sigma^2 = p(1-p) \approx p$ 成立, 是因为对于大部分的二元组来说 p 很小。

在语料库的 14 307 668 个二元组中,实际上 *new companies* 出现了 8 次。所以,对于这个样本,我们得到的样本均值是 $\bar{x} = \frac{8}{14\,307\,668} \approx 5.591 \times 10^{-7}$ 。现在我们有在 t 检验中使用的所有数据:

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}} \approx \frac{5.591 \times 10^{-7} - 3.615 \times 10^{-7}}{\sqrt{\frac{5.591 \times 10^{-7}}{14\,307\,668}}} \approx 0.999\,932$$

t 值为 0.999 932 小于 2.576, 而 2.576 是 $\alpha = 0.005$ 的临界值, 所以不能拒绝零假设: *new* 和 *companies* 是独立出现的, 它们不能形成搭配。这似乎是一个正确的结果: 短语 *new companies* 完全符合组合构词法, 这里并且没有额外的含义成分可以把它提升为搭配。这个 t 值令人惊异地接近 1.0, 但是这只是一个巧合。参见习题 5.5。

表 5.6 给出了 10 个二元组的 t 值, 这些二元组在我们的语料库中出现了 20 次。对于前 5 个二元组, 在 $\alpha = 0.005$ 的置信水平下我们可以拒绝“这些组成词是独立出现的”的零假设, 所以这些二元组是可以作为搭配的上佳候选。下面 5 个二元组在置信度下测试是失败的, 所以我们不会把它们作为搭配的候选。

可以看到, 基于频率的方法不能对这 10 个二元组排序, 因为它们出现的频率是一样的。考虑一下表 5.6 中的计数, 可以看到 t 检验考虑了二元组 ($C(w_1 w_2)$) 出现次数相对于它们的组成词的出现频率的比例。如果两个词同现的次数相对于其中每个词的出现频率所占比例较高 (*Ayatollah Ruhollah, videocassette recorder*), 或者至少相对于二元组中一个词的出现频率所占的比例非常高 (*unsalted*), 那么它的 t 值就会是高的。这个判断标准和人的直观印象一致。

表 5.6 搭配发现:把 t 检验应用到出现频率为 20 的 10 个二元组上

t	$C(w^1)$	$C(w^2)$	$C(w^1 w^2)$	w^1	w^2
4.4721	42	20	20	Ayatollah	Ruhollah
4.4721	41	27	20	Bette	Midler
4.4720	30	117	20	Agatha	Christie
4.4720	77	59	20	videocassette	recorder
4.4720	24	320	20	unsalted	butter
2.3714	14 907	9017	20	first	made
2.2446	13 484	10 570	20	over	many
1.3685	14 734	13 478	20	into	them
1.2176	14 093	14 776	20	like	people
0.8036	15 019	15 629	20	time	last

和本章大部分的内容不一样,表 5.6 的分析中包括了一些停止词,这是因为如果没有停止词很难找到置信失败的例子。事实上,在我们的语料库中大部分二元组出现的概率要大于它们偶然出现的概率。在语料库中出现 20 次的 831 个二元组中,有 824 个的置信度的零假设是可以被拒绝的。但是我们仅仅把其中的很少一部分作为真正的搭配。可能不独立的二元组 $\left(\frac{824}{831} \approx 0.99\right)$ 的比例如此之高的原因是,和随机的词汇生成器相比较,语言是非常规则的,以至于只有很少的事件是完全不可预测的。更进一步说,搭配只是我们解决语言问题(比如语义消歧和概率句法分析)的基础,这些语言问题将在其他章节讨论。 t 检验和其他一些统计测试对于搭配可能性排序是非常有用的方法,而置信水平本身的用处不大。事实上,在本章引用的大部分文献中,置信水平从来没有被考虑过,它们都使用的是打分和结果排序的方法。

5.3.2 差异的假设检验

t 检验也可以用来解决有细微差异的搭配发现问题:寻找一些词,它们的同现模式可以最好地区分两个意义相近词之间的差别。例如,在计算词典编撰法中,我们希望找到这样的一些词,它们可以很好地区分 *strong* 和 *powerful* 的不同。Church and Hanks(1989)提出使用 t 检验来解决这个问题。表 5.7 给出了与 *strong* 相比较最经常和 *powerful* 同现的 10 个词(前 10 个词),以及与 *powerful* 相比较,最经常和 *strong* 同现的 10 个词(后 10 个词)。

t 的数值使用下面的公式计算,这里把 t 检验扩展到了两个正态分布均值的比较:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \quad (5.4)$$

这里的零假设是,平均差异为 0 ($\mu = 0$),所以我们得到 $\bar{x} - \mu = \bar{x} = \frac{1}{N} \sum (\mathcal{X}_{1i} - \mathcal{X}_{2i}) = \bar{x}_1 - \bar{x}_2$ 。分母是两个样本的方差相加之和,因为两个独立的随机变量差异的方差是它们单独方差的和。

现在我们可以解释表 5.7 中的内容了。假设样本符合 Bernoulli 分布, t 的数值可以被计算出来(就像我们首先介绍的 t 检验的基本版本一样)。如果 w 是我们感兴趣的搭配词(例如 *computers* 或 *symbol*), v^1 和 v^2 是要比较的词(例如 *powerful* 和 *strong*),那么可以得到 $\bar{x}_1 = s_1^2 = P(v^1 w)$, $\bar{x}_2 = s_2^2 = P(v^2 w)$ 。这里再一次使用了 $s^2 = p - p^2 \approx p$ 的近似:

$$t \approx \frac{P(v^1w) - P(v^2w)}{\sqrt{\frac{P(v^1w) + P(v^2w)}{N}}}$$

可以把这个公式简化如下:

$$\begin{aligned} t &\approx \frac{\frac{C(v^1w)}{N} - \frac{C(v^2w)}{N}}{\sqrt{\frac{C(v^1w) + C(v^2w)}{N^2}}} \\ &= \frac{C(v^1w) - C(v^2w)}{\sqrt{C(v^1w) + C(v^2w)}} \end{aligned} \quad (5.5)$$

在公式中, $C(x)$ 表示 x 在语料库中出现的次数。

表 5.7 经常与 *powerful* 同现的词(前 10 个词)或经常与 *strong* 同现的词(后 10 个词)

t	$C(w)$	$C(\text{strong } w)$	$C(\text{powerful } w)$	单词
3.1622	933	0	10	computers
2.8284	2337	0	8	computer
2.4494	289	0	6	symbol
2.4494	588	0	6	machines
2.2360	2266	0	5	Germany
2.2360	3745	0	5	nation
2.2360	395	0	5	chip
2.1828	3418	4	13	force
2.0000	1403	0	4	friends
2.0000	267	0	4	neighbor
7.0710	3685	50	0	support
6.3257	3616	58	7	enough
4.6904	986	22	0	safety
4.5825	3741	21	0	sales
4.0249	1093	19	1	opposition
3.9000	802	18	1	showing
3.9000	1641	18	1	sense
3.7416	2501	14	0	defense
3.6055	851	13	0	gains
3.6055	832	13	0	criticism

由 (Church and Hanks 1989) 建议的这种 t 检验形式应用称为词典编撰法。表 5.7 中的数据对于词典编撰者来说是有用的, 他们希望编撰准确的辞典条目, 能够说明 *strong* 和 *powerful* 之间的不同。基于一些有效的搭配, Church and Hanks 分析了搭配内部和外部特性的区别。例如, 句子 *strong support from a demographic group* 中对 *strong* 的意思表明, 虽然这个人口普查小组对统计工作很投入, 但是他们可能没有任何决策权力。所以 *strong* 刻画的是一种内部的特性。相反, 一个 *powerful supporter* 表示了一个有权力改变事情的人。在我们的语料库中可以找到很多这样的搭配, 支持 Church and Hanks 的分析。然而两个词汇的语义差异有更加复杂的情况, 因为词的内部和外部特性有时依赖于很微妙的东西, 比如文化背景。例如, 一方面我们说 *strong tea*, 另一方面我们说 *powerful drugs*; 两种说法告诉我们, 我们对 *tea* 和 *drugs* 的态度的差异要比两个形容词的语义的差异大, 参见 (Church et al. 1991: 133)。

5.3.3 皮尔逊卡方检验

t 检验的缺陷在于它假设了数据的先验分布为正态分布,而一般情况下该假设和真实数据分布并不一致,参见(Church and Mercer 1993:20)。在验证数据是否独立时, χ^2 检验通常可作为 t 检验的一种替代检验方法。与 t 检验不同,它不要求数据满足正态分布。最简单的情况下, χ^2 检验可应用于具有表 5.8 形式的 2×2 的表。统计检验的实质在于对比表中的观测频度和期望频度,以验证是否独立。当它们之间的差别很大时,可以否定独立的零假设。

表 5.8 给出了前面所用语料库中两个英文单词 *new* 和 *companies* 的分布关系。它们的出现次数分别为 $C(\text{new}) = 15\,828$, $C(\text{companies}) = 4675$, $C(\text{new companies}) = 8$, 语料库中共有 14 307 668 个词次(token)。由此可以计算出语料库中第一个单词不是 *new* 而第二个单词为 *companies* 的两字序列共有 $4667 = 4675 - 8$ 。表中下面一行的统计数字可以通过类似的方法得出。

χ^2 统计量计算了观测值和期望值之间差别的总和,并且将期望值作为比例因子,得到:

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \quad (5.6)$$

其中 i 表示表中的行变量, j 表示列变量, O_{ij} 表示表单元 (i, j) 的观测值, E_{ij} 表示期望值。

很容易看出 χ^2 是渐近 χ^2 分布的,换句话说,当数值很大时, χ^2 满足 χ^2 分布。在稍后的章节中我们对这种近似程度进行讨论。

通过计算边缘分布可以得到期望频度 E_{ij} 的值,计算方法不复杂,将表中频度值转化为比例值后分别按行和列计算总数即可。举例来说,表单元 $(1, 1)$ (*new companies*) 的期望频率就是计算 *new companies* 中单词 *new* 作为二元对的第一个词的边缘分布乘以 *companies* 作为二元对的第二个词的边缘分布,再乘以语料库中二元对的总数。

$$\frac{8 + 4667}{N} \times \frac{8 + 15\,820}{N} \times N \approx 5.2$$

这就意味着当单词 *new* 和 *companies* 相互完全独立分布时, *new companies* 在我们假设的语料库的每篇文本中平均期望出现次数为 5.2。

表 5.8 表明单词 *new* 和 *companies* 出现次数之间的依赖关系的 2×2 的表

	$w_1 = \text{new}$	$w_1 \neq \text{new}$
$w_2 = \text{companies}$	8 (<i>new companies</i>)	4667 (例如 <i>old companies</i>)
$w_2 \neq \text{companies}$	15 820 (例如 <i>new machines</i>)	14 287 173 (例如 <i>old machines</i>)

从表 5.8 中我们可以看到,在语料库中 *new companies* 一共出现 8 次, 4667 个二元组的第二个单词为 *companies*, 但第一个单词不是 *new*。而第一个单词为 *new*, 第二个单词不为 *companies* 的二元组共有 15 820 个。14 287 181 个二元组既不含单词 *new*, 也不含 *companies*。

χ^2 检验从理论上讲适用于各种大小的表,但是对于 2×2 的表的表达形式相对简单(参见习题 5.9):

$$\chi^2 = \frac{N(O_{11}O_{22} - O_{12}O_{21})^2}{(O_{11} + O_{12})(O_{11} + O_{21})(O_{12} + O_{22})(O_{21} + O_{22})} \quad (5.7)$$

将表 5.8 中的值代入式(5.7),可以得到以下 χ^2 数值:

$$\frac{14\,307\,668(8 \times 14\,287\,173 - 4667 \times 15\,820)^2}{(8 + 4667)(8 + 15\,820)(4667 + 14\,287\,173)(15\,820 + 14\,287\,173)} \approx 1.55$$

通过查找附录 A 中的 χ^2 分布表,会发现当置信水平为 $\alpha = 0.05$ 时,临界值 $\chi^2 = 3.841$ (假设 2×2 的表的分布的自由度为 1)。根据这个值,我们可以接受单词 *new* 和 *companies* 之间的独立性假设,所以词序列 *new companies* 不能作为一个固定搭配。这个结果和我们由 t 检验得到的结果是一致的。

通常来说,当检验单词的固定搭配时, t 统计和 χ^2 统计的结果差别并不是很大。例如,对所有两字词序列计算 t 分布和 χ^2 分布的统计值并按其大小排列,会发现最前面的 20 个两字词序列其实是相同的。

但是,和 t 检验不同, χ^2 检验在概率值比较大的情况下同样适用,这种情况往往不符合 t 检验要求的正态分布,这也是它广泛应用于词的搭配问题的原因。

在统计语言处理中较早应用 χ^2 统计量的是在统计机器翻译领域,主要用来确定对齐语料库中的翻译对(church and Gale 1991b^①)。表 5.9 中的数据就清楚地表明法语单词 *vache* 对应于英语单词 *cow*。表中数值 59 表示在对齐的英语句子和法语句子中,单词 *cow* 和 *vache* 的同现次数。这时 χ^2 具有很高的值: $\chi^2 = 456\,400$,所以我们以很高的置信度确认单词 *cow* 和 *vache* 的相互独立性的零假设是不成立的,将法文单词 *vache* 译成 *cow* 是合理的。

表 5.9 对齐语料库中单词 *vache* 和 *cow* 的对照。在表中使用 χ^2 检验可以确定 *vache* 和 *cow* 是否互为翻译

	<i>cow</i>	$\neg$ <i>cow</i>
<i>vache</i>	59	6
$\neg$ <i>vache</i>	8	570 934

另外一个很有意思的问题就是, χ^2 统计量可以看成衡量不同语料库之间相似度的指标(Kilgariff and Rose 1998)。我们可以编制 $n \times 2$ 大小的表,比如取 n 的值为 500。那么表中的列对应于两个不同的语料库,而行和语料中的词一一对应。以表 5.10 为例,因为表中的比率值大致都为 6,所以可以认为表中两个语料库的相似度比较高,内容上是同构的。另一方面,如果比率值的变化很大,语料库 χ^2 之间异构的可能性就比较大。

表 5.10 使用 χ^2 检验不同语料库之间词语的相似性。
这样的检验可以衡量语料库之间的相似性

	语料库1	语料库2
词1	60	9
词2	500	76
词3	124	20
...	...	...

我们知道 t 检验的一个问题在于它的前提假设,它认为数据满足正态分布,而 χ^2 的不足在于当统计出的数值很小时,结果的说服力不是很强。Snedecor and Cochran(1989:127)建议,如果样本容量小于 20 或者尽管在 20 和 40 之间但是表单元的期望值小于 5,最好不应用 χ^2 统计量。

① 他们实际上使用了一种称为 ϕ^2 的度量,即 χ^2 与 N 的乘积。之所以这样做,是因为他们只对翻译对的排名感兴趣,所以评估意义不是特别重要。

5.3.4 似然比

似然比是假设检验的另一种方法。在下面将会看到,对于稀疏数据,它比 χ^2 检验更加合适,而且还有这样的优点:计算出来的似然比统计值比 χ^2 检验的统计值更有可解释性。它是一个简单的数字,可以告诉我们一个假设的可能性比其他假设大多少。

在应用似然比检验到搭配发现的过程中,我们考察用下面两个可选的假设来解释二元组 w_1w_2 的出现频率,参见(Dunning 1993)的假设:

- 假设1: $P(w^2|w^1) = p = P(w^2|\neg w^1)$
- 假设2: $P(w^2|w^1) = p_1 \neq p_2 = P(w^2|\neg w^1)$

假设1是独立性假设的形式化(w_2 的出现和前面 w_1 的出现是独立的),假设2是非独立性假设的形式化,对于我们感兴趣的搭配,它是一个很好的证据^①。

使用最大似然估计的方法计算 p, p_1 和 p_2 ,用 c_1, c_2 和 c_{12} 来表示在语料库中 w_1, w_2 和 w_1w_2 的出现次数:

$$p = \frac{c_2}{N} \quad p_1 = \frac{c_{12}}{c_1} \quad p_2 = \frac{c_2 - c_{12}}{N - c_1} \quad (5.8)$$

假设二项式分布:

$$b(k; n, x) = \binom{n}{k} x^k (1-x)^{(n-k)}. \quad (5.9)$$

实际观测到的 w_1, w_2 和 w_1w_2 频率的似然值是 $L(H_1) = b(c_{12}; c_1, p)b(c_2 - c_{12}; N - c_1, p)$ (假设1的情况)和 $L(H_2) = b(c_{12}; c_1, p_1)b(c_2 - c_{12}; N - c_1, p_2)$ (假设2的情况)。表5.11中总结了这些讨论。得到的似然值 $L(H_1)$ 和 $L(H_2)$ 是最后两行的乘积,并且它们分别对应于 w_1w_2 和 $\neg w_1w_2$ 的出现频率。

表 5.11 如何计算 Dunning 的似然比检验。例如, H_2 的似然值是最右边一列的最后两行的乘积

	H_1	H_2
$P(w^2 w^1)$	$p = \frac{c_2}{N}$	$p_1 = \frac{c_{12}}{c_1}$
$P(w^2 \neg w^1)$	$p = \frac{c_2}{N}$	$p_2 = \frac{c_2 - c_{12}}{N - c_1}$
条件 c_1 下 c_{12} 对应的二元组是 w^1w^2	$b(c_{12}; c_1, p)$	$b(c_{12}; c_1, p_1)$
条件 $N - c_1$ 下 $c_2 - c_{12}$ 对应的二元组是 $\neg w^1w^2$	$b(c_2 - c_{12}; N - c_1, p)$	$b(c_2 - c_{12}; N - c_1, p_2)$

似然比 λ 的对数值如下:

$$\begin{aligned}
 \log \lambda &= \log \frac{L(H_1)}{L(H_2)} \\
 &= \log \frac{b(c_{12}; c_1, p)b(c_2 - c_{12}; N - c_1, p)}{b(c_{12}; c_1, p_1)b(c_2 - c_{12}; N - c_1, p_2)} \\
 &= \log L(c_{12}, c_1, p) + \log L(c_2 - c_{12}, N - c_1, p) \\
 &\quad - \log L(c_{12}, c_1, p_1) - \log L(c_2 - c_{12}, N - c_1, p_2)
 \end{aligned} \quad (5.10)$$

公式中的 $L(k, n, x) = x^k (1-x)^{n-k}$ 。

把似然比检验应用到纽约时报语料库(语料库的 $N = 14\,307\,668$)中,表5.12列出了20个

① 我们假设,如果假设2成立,那么 p_1 远大于 p_2 。在这里 p_1 远小于 p_2 的情况是很少的,所以将其忽略了。

排在最前面的含有 *powerful* 的二元组,这是按照似然比的大小进行排列的。下面会进一步解释为什么用 $-2\log\lambda$ 代替 λ 。在这里可以考虑出现的所有二元组,包括那些稀有的二元组(出现次数少于6次),因为这种检验对于稀有的二元组也很有效。例如, *powerful cudgels* 只出现了两次,但是也被识别为一个可能的搭配。

使用似然比检验的一个优点是,它有一个很清晰直观的解释。例如,二元组 *powerful computers* 的似然比是 $e^{0.5 \times 82.96} \approx 1.3 \times 10^{18}$,这表明它非常可能符合假设 2。*computers* 出现在 *powerful* 的后面比 *computers* 偶然出现的可能性更大,也就是说,这两个词的出现是互相依赖的。似然比的数值比我们查表得到的 t 检验和 χ^2 检验的数值更容易理解。

似然比检验的方法的另一个很重要的优点是,它比 χ^2 检验更好地解决了稀疏数据问题。怎样把似然比应用到假设测试上呢?如果说 λ 是一个具体形式的似然比,那么 $-2\log\lambda$ 渐进逼近 χ^2 分布,参见(Mood et al. 1974:440)。所以我们可以使用表 5.12 中的值来测试两个可选的零假设 H_1 和 H_2 。例如,我们可以在表中查到 *powerful cudgels* 的值是 34.15,并且对于这个二元组,在置信水平 $\alpha = 0.005$ 的情况下可以拒绝 H_1 。临界值是 7.88(自由度为 1)。参见附录 A 中的 χ^2 分布表。

我们需要的似然比的具体形式是子参数空间最大似然估计与全参数空间最大似然估计的比值。对于表 5.10 中的似然比,全空间 (p_1, p_2) 是关于 w_1 先出现时 w_2 出现的概率(p_1)和其他词先出现时 w_2 出现的概率(p_2)的两维空间。如果我们假设最大似然估计就像式(5.8)中计算的那样,可以得到观测数据的最大似然值。参数的子空间是 $p_1 = p_2$ 情况的子集。利用式(5.8)计算观测数据子空间上的最大似然值,可以看到,如果 λ 是这两个似然值的比值(一个是在子空间上的最大似然值,一个是在全空间上的最大似然值),那么 $-2\log\lambda$ 是渐进逼近 χ^2 分布的。“渐进”的大致意思是“如果数字足够大,那么它们的值就是近似相等的”。在具体情况下,数字是否足够大是很难判断的,但是 Dunnig 已经证明,对于小的统计数字,表 5.11 的似然比近似到 χ^2 分布也要好于表 5.6 的 χ^2 的统计值。因此,一般说来,对于搭配发现,似然比检验要比 Pearson 的 χ^2 检验更适合^①。

相对频率比 迄今为止,我们已经在在一个语料库的范围内考察了搭配存在的证据。两个或者多个不同语料库的相对频率的比值也可以帮助我们发现搭配,在语料之间进行比较的时候,这些搭配可以反映一个语料库的特点(Damerau 1993)。虽然相对频率比不能很好地融入我们设计的假设检验的范例中,但是在这里我们仍然关注它们,因为它们可以被解释为似然比。

表 5.13 显示了在我们的参考语料库 1990 年的纽约时报中出现 2 次的 10 个二元组。这些二元组按照它们在 1989 年和 1990 年语料中(都是从 8 月到 11 月的语料)的出现频率的相对频率比进行排列。例如, Karim Obeid 在 1989 年的语料中出现 68 次。所以它的相对频率比 r 是:

$$r = \frac{\frac{2}{14\,307\,668}}{\frac{68}{11\,731\,564}} \approx 0.024\,116$$

在表 5.13 中列出的二元组大部分对应于某个新闻条目,它们在 1989 年要比 1990 年流行。这些新闻是:穆斯林传教士(Muslim cleric) *Sheik Abdul "Karim Obeid"* (他在 1989 年被绑架),东欧共产主义的解体(*East Berliners, EAST GERMANS, Prague Spring*), *John Le Carre* 的小说 "*The Russia House*",住房供给和城市发展部门的丑闻(HUD),还有 San Francisco 海湾地区 10 月

① 但是,如果 2×2 的关联表(contingency table)中的期望值小于 1.0,即使是 $-2\log\lambda$ 对 χ^2 的近似也不是很好,见(Cressie 1988; Pedersen 1996)。

“17 earthquake” (17 日的地震)。但是我们也发现了一些假的搭配,例如 *Miss Manners* (他在纽约时报的新闻专栏在 1990 年停止) 和 *Among individual*。报告者 Phillip H. Wiggins 喜欢在他的股票分析中使用后面的短语 (*Among individual Big Board issues ...*), 但是他也在 1990 年停止了写作。

表 5.12 按照 Dunning 的似然比检验方法, 得分最高的前 20 个含有 *powerful* 的二元组

$-2 \log \lambda$	$C(w^1)$	$C(w^2)$	$C(w^1 w^2)$	w^1	w^2
1291.42	12 593	932	150	most	powerful
99.31	379	932	10	politically	powerful
82.96	932	934	10	powerful	computers
80.39	932	3424	13	powerful	force
57.27	932	291	6	powerful	symbol
51.66	932	40	4	powerful	lobbies
51.52	171	932	5	economically	powerful
51.05	932	43	4	powerful	magnet
50.83	4458	932	10	less	powerful
50.75	6252	932	11	very	powerful
49.36	932	2064	8	powerful	position
48.78	932	591	6	powerful	machines
47.42	932	2339	8	powerful	computer
43.23	932	16	3	powerful	magnets
43.10	932	396	5	powerful	chip
40.45	932	3694	8	powerful	men
36.36	932	47	3	powerful	486
36.15	932	268	4	powerful	neighbor
35.24	932	5245	8	powerful	political
34.15	932	3	2	powerful	cudgels

表 5.13 Damerau 的频率比测试。在 1990 年的纽约时报语料库中出现 2 次的 10 个二元组, 它们的排列顺序 (倒序) 是按照在 1989 年和 1990 年语料中的出现频率的相对频率比确定的

比值	1990	1989	w^1	w^2
0.0241	2	68	Karim	Obeid
0.0372	2	44	East	Berliners
0.0372	2	44	Miss	Manners
0.0399	2	41	17	earthquake
0.0409	2	40	HUD	officials
0.0482	2	34	EAST	GERMANS
0.0496	2	33	Muslim	cleric
0.0496	2	33	John	Le
0.0512	2	32	Prague	Spring
0.0529	2	31	Among	individual

这些例子显示, 频率比主要用来发现那些特殊主题的搭配。Damerau 设计的这个应用是为了比较一般文本和特殊主题文本。这些在特殊主题文本中更经常出现的词汇和短语很可能是特殊领域词表的一部分。

5.4 互信息

对于搭配发现,一种以信息论为根据的方法是点互信息(pointwise mutual information),参见(Church et al. 1991; Church and Hanks 1989; Hindle 1990)。(Fano 1961: 27 ~ 28)最初定义了两个特殊事件 x' 和 y' 之间的互信息,对于我们的情况,两个具体词同现的互信息如下:

$$I(x', y') = \log_2 \frac{P(x' y')}{P(x') P(y')} \quad (5.11)$$

$$= \log_2 \frac{P(x' | y')}{P(x')} \quad (5.12)$$

$$= \log_2 \frac{P(y' | x')}{P(y')} \quad (5.13)$$

我们在 2.2.3 节介绍的这种互信息形式,粗略地说是一种度量形式,表明一个词和其他词语有多大的联系,我们将会对它进行更加精确的描述。

在信息论中,互信息经常被定义成随机变量之间的联系,而不是在这里定义的随机变量的值(参见 2.2.3 节的标准定义)。下面将会看到,这两种类型的互信息是完全不同的。

当把这个定义应用到表 5.6 中的 10 个搭配时,我们得到了和使用 t 检验结果一样的排列(参见表 5.14)。和通常一样,我们使用最大似然估计来计算概率,例如:

表 5.14 搭配发现:按照互信息大小进行排列的出现 20 次的 10 个二元组

$I(w^1, w^2)$	$C(w^1)$	$C(w^2)$	$C(w^1 w^2)$	w^1	w^2
18.38	42	20	20	Ayatollah	Ruhollah
17.98	41	27	20	Bette	Midler
16.31	30	117	20	Agatha	Christie
15.94	77	59	20	videocassette	recorder
15.19	24	320	20	unsalted	butter
1.09	14 907	9017	20	first	made
1.01	13 484	10 570	20	over	many
0.53	14 734	13 478	20	into	them
0.46	14 093	14 776	20	like	people
0.29	15 019	15 629	20	time	last

$$I(\text{Ayatollah}, \text{Ruhollah}) = \log_2 \frac{\frac{20}{14\,307\,668}}{\frac{42}{14\,307\,668} \times \frac{20}{14\,307\,668}} \approx 18.38$$

那么确切地说,什么是(点)互信息 $I(x', y')$,它是一种什么样的度量呢? Fano 的互信息定义是(见式(5.12)):

由于事件 $[y']$ 发生与事件 $[x']$ 发生相关联而提供的信息量,如式(5.12)所定义。

例如,互信息的方法可以告诉我们这样的信息量,如果知道了 Ruhollah 在语料库中的 $i+1$ 位置出现,那么 Ayatollah 在语料库中的 i 位置出现的信息量就增加了 18.38 比特。或者,因为式(5.12)和式(5.13)是相等的,它也同样告诉我们这样的信息量,如果知道了 Ayatollah 在语料库中的 i 位置出现,那么 Ruhollah 在语料库中的 $i+1$ 位置出现的信息量就增加了 18.38 比特。

也可以这样说,不确定性减少了 18.38 比特。换句话说,如果我们被告知 Ayatollah 是当前词,我们能够更有把握地说, Ruhollah 将会在下一个位置出现。

遗憾的是,在很多情况下“信息增加”的度量并不是一种很好的度量方法,它无法衡量两个事件之间的联系是什么,就像很多研究者已经指出的一样。在这里,我们讨论的基础主要是 (Church and Gale 1991b) 和 (Maxwell 1992)。考虑表 5.15 中的两个例子,它们是 Hansard 语料库中英语和法语句子的相对应词的频率,这个语料库是加拿大国会辩论的双语对齐语料库(和表 5.9 类似)。House 经常出现在包含 *chambre* 和 *communes* 的法语句子的翻译中的原因是,在 Hansard 语料库中, *house* 最经常的用法是 *House of Commons*, 这个短语对应的法语翻译是 *Chambre de communes*。但是很容易看到, *communes* 相对于 *chambre* 是一个错误的翻译,因为 *house* 出现的大多数情况对应的法语没有 *communes*。就像表中数据显示的一样, χ^2 测试能够推断出正确的对应,但是互信息给出了错误的结果 (*communes/house*)。

如果我们考虑互信息的定义式 (5.12) 并比较数值 $I(\text{chambre}, \text{house})$ 和 $I(\text{communes}, \text{house})$, 就能很容易地解释这两种度量方法的不同:

$$\begin{aligned} \log \frac{P(\text{house}|\text{chambre})}{P(\text{house})} &= \log \frac{\frac{31\ 950}{31\ 950+4793}}{P(\text{house})} \approx \log \frac{0.87}{P(\text{house})} \\ &< \log \frac{0.92}{P(\text{house})} \approx \log \frac{\frac{4974}{4974+441}}{P(\text{house})} = \log \frac{P(\text{house}|\text{communes})}{P(\text{house})} \end{aligned}$$

在法语中出现 *communes*, 相应地在英语中出现 *house* 的可能性要比在法语中出现 *chambre*, 相应地在英语中出现 *house* 的可能性大。对于 *communes* 的比较大的互信息反映了这样的事实, *communes* 引起了较大的不确定性的减少。相反, χ^2 是一个直接概率依赖性的检验,在这个上下文中我们可以把它解释成两个词之间联系的程度,因此它可以用来检验词对是不是翻译对和搭配。

表 5.15 在对齐 Hansard 语料库中, *chambres/house*, *communes/house* 之间的对应。对于 *communes/house*, 互信息给出了一个比较高的分数,但是 χ^2 检验对正确的翻译对 *chambre/house* 的分数较高

	<i>chambre</i>	$\neg$ <i>chambre</i>	MI	χ^2
<i>house</i>	31 950	12 004	4.1	553 610
$\neg$ <i>house</i>	4793	848 330		
	<i>communes</i>	$\neg$ <i>communes</i>		
<i>house</i>	4974	38 980	4.2	88 405
$\neg$ <i>house</i>	441	852 682		

表 5.16 中显示了使用互信息来发现搭配的第二问题。我们列出了 10 个二元组,它们在参考语料库中前 1000 个文档中只出现了一次,并且它们的互信息也是基于这 1000 个文档的。在表的右半部分显示了它们基于全部语料库的互信息(大约有 23 000 个文档)。

这个有 23 000 个文档的更大的语料库有可能使我们做出更好的估计,但是它也仅仅导致了排列的稍微优化。二元组 *marijuana growing* 和 *new converts* (可能的搭配)上移, *Reds survived* (肯定不是搭配)下移。然而有一点很明显,即使我们使用一个 10 倍大的语料库,有 6 个二元组仍然只出现一次,因此它们的最大似然估计是不准确的,并且人为地夸大了它们的互信息的值。所有的这 6 个二元组都不是搭配,我们应该使用更好的方法来排列它们。

表 5.16 在稀疏数据中使用互信息的问题。表中显示了 10 个二元组,它们在参考语料库中的前 1000 个文档中出现了一次,并且分别按照在前 1000 个文档中的互信息大小(见表的左半部分),在全部语料库中的互信息大小(见表的右半部分)来排列。这些例子证明,大部分的二元组不能被语料数据很好地刻画(即使是很大的语料库),而且互信息对于数据稀疏带来的不准确非常敏感

I_{1000}	w^1	w^2	$w^1 w^2$	二元组	I_{23000}	w^1	w^2	$w^1 w^2$	二元组
16.95	5	1	1	Schwartz eschews	14.46	106	6	1	Schwartz eschews
15.02	1	19	1	fewest visits	13.06	76	22	1	FIND GARDEN
13.78	5	9	1	FIND GARDEN	11.25	22	267	1	fewest visits
12.00	5	31	1	Indonesian pieces	8.97	43	663	1	Indonesian pieces
9.82	26	27	1	Reds survived	8.04	170	1917	6	marijuana growing
9.21	13	82	1	marijuana growing	5.73	15 828	51	3	new converts
7.37	24	159	1	doubt whether	5.26	680	3846	7	doubt whether
6.68	687	9	1	new converts	4.76	739	713	1	Reds survived
6.00	661	15	1	like offensive	1.95	3549	6276	6	must think
3.81	159	283	1	must think	0.41	14 093	762	1	like offensive

没有一种我们已知的方法可以很好地解决低频率事件,但是可以证明,数据稀疏问题对于互信息是一个很困难的难题。应该注意到,互信息是二元组 $P(w_1 w_2)$ 的概率和两单独词的概率乘积 $P(w_1)P(w_2)$ 的似然对数比。考虑两种极端的情况:两个词的出现是完全互相依赖的(它们都是一起出现)或完全独立的(一个词出现不能给出关于其他词出现的任何信息)。对于完全的互相依赖,有:

$$I(x, y) = \log \frac{P(xy)}{P(x)P(y)} = \log \frac{P(x)}{P(x)P(y)} = \log \frac{1}{P(y)}$$

也就是说,在完全依赖的二元组中,当它们出现的次数减少时,它们的互信息增加。

对于完全独立的情况,有:

$$I(x, y) = \log \frac{P(xy)}{P(x)P(y)} = \log \frac{P(x)P(y)}{P(x)P(y)} = \log 1 = 0$$

我们可以说,互信息是衡量独立性的一种很好的方法。接近 0 的互信息值表明了独立性(概率的独立性)。但是互信息不是衡量依赖性的一种很好的方法,因为对于依赖性来说,互信息的值是由单独词的频率决定的。在其他条件相等的情况下,由低频率词组成的二元组的互信息要大于高频率词组成的二元组。这一点是我们不希望看到的,因为高频率意味着有更多的信息,并且对于那些有更多我们感兴趣的信息的二元组,我们宁愿它们的排列更靠前。对于这个问题已经设计出的一种解决方法是使用截断(cutoff),只考虑出现频率大于 3 的词。但是这种变动不能从根本上解决问题,只是改善它的效果而已。由于点互信息对于我们发现感兴趣的搭配的效果并不明显,所以通常在一些具体应用中不使用它,参见 (Fontenelle et al. 1994:81), 或者重新定义为 $C(w_1 w_2) I(w_1 w_2)$, 来纠正原定义对低频事件的偏爱,参见 (Fontenelle et al. 1994:72, Hodges et al. 1996)。

如前所述,这里使用的互信息定义在语料库研究中经常使用,但是在信息论中不常用。信息论中的互信息是指本节使用的互信息的期望值:

$$I(X; Y) = E_{p(x,y)} \log \frac{p(X, Y)}{p(X) p(Y)}$$

我们在本章使用的互信息定义是一个比较老的版本,称为点互信息,参见 2.2.3 节(Fano 1961: 28, Gallager 1968)。表 5.17 总结了新旧定义的命名惯例。其中一个是另一个的期望值,因此这两种互信息是非常不同的。

表 5.17 (Cover and Thomas 1991)和(Fano 1961)中的互信息的不同定义

表示符号	定义	通常使用的名称	Fano 使用的名称
$I(x, y)$	$\log \frac{p(x, y)}{p(x)p(y)}$	点互信息	互信息
$I(X; Y)$	$E \log \frac{p(X, Y)}{p(X)p(Y)}$	互信息	平均互信息/互信息的期望值

互信息的例子证明了一个很简单的道理:考察一个形式化的数学概念是很重要的。我们在这里使用的点互信息($\log \frac{p(w_1 w_2)}{p(w_1)p(w_2)}$)概念衡量了知道一些其他词出现时某个词出现的不确定性的减少。正如本节所述,这种方法对于获取语言属性的作用是有限的。

5.5 搭配的概念

没有语言学背景的读者可能会混淆搭配的概念。我们将用这一节篇幅更详细地讨论搭配是什么。

事实上,搭配概念存在很多不同的定义。一些计算和统计学文献的作者把搭配定义为:带有特殊性质的两个或多个连续的词,例如 Choueka(1988):

搭配的定义 两个或多个连续的词序列,具有句法和语义单位的特性,并且它的准确无歧义的意思或含义不能直接由它的组成部分的意思和含义直接得出。

在本章中已经给出的很多例子中假设了词之间是相邻的,但是在大部分语言学的主导研究中两个词甚至不连续也可以是一个搭配(例如 *knock... door*)。下面的标准是典型语言学对搭配的衡量标准,参见 Benson(1989)和 Brundage et al.(1992)。其中,非组成构词法是我们的主要依据。

- **非组成构词法 (Non-compositionality)**: 搭配的含义不是它的各个组成部分含义的简单组合。或者是完全不同于自由组合而得出的含义(例如成语 *kick the bucket*),或者是有另外的含义或附加的意思,这些含义和意思不能从它的组成部分中预测出来。例如, *white wine*, *white hair* 和 *white woman* 都是指一些稍微不同的颜色,所以认为它们是搭配。
- **不可替换性 (Non-substitutability)**: 我们不能用其他词来替换搭配成分,即使在上下文中它们的意思相同也不行。例如,我们不能用 *yellow wine* 来代替 *white wine*,即使 *yellow* 像 *white* 一样能够很好地描述 *white wine* 中的颜色(这是一种带有浅黄色的白色)。
- **不可更改性 (Non-modifiability)**: 很多搭配不能通过额外的词汇或者语法转换进行自由的修改。对于像成语这样的固定表达尤其如此。例如,即使一般像 *frog* 这样的名词可以被形容词 *ugly* 修饰,我们也不能把 *to get a frog in one's throat* 改成 *to get an ugly frog in one's throat*。类似地,从单数变为复数也会产生错误的成语,例如 *people as poor as church mice*。

对于一个组合是否是搭配这样的问题,一种好的检验方法是把它翻译成其他语言。如果我们不能逐词翻译,那么很可能这个组合就是一个搭配结构。例如,把 *make a decision* 逐词翻译成法语,得到的是 *faire une décision*,这是错误的。在法语中应该说 *prendre une décision*。所以可以确定, *make a decision* 在英语中是一个搭配。

一些作者甚至更进一步推广了搭配的概念,即包括了互相之间有很强联系的词,这些词不必一起出现在一个语法单元中,也不必有特殊的次序,例如 *doctor-nurse* 和 *plane-airport* 这样的情况。把搭配的概念限制在一个比较狭义的语法范畴中可能会更好一些,对于那些在相同的上下文中经常出现的词汇现象,我们可以使用术语“关联”(association)和“同现”(co-occurrence)来描述它们。

考查搭配类型是有益的,这是一种纯粹的文本语言学分析任务。如果时间和人力资源是充分的,那么我们就可以不考虑统计分析和计算技术的局限性。一个纯粹的语言分析的的例子是“BBI Combinatory Dictionary of English”(Benson et al. 1993)。在表 5.18 中列出了词典中 *strength* 和 *power* 的一些搭配(或者组合,因为词典更倾向于这样称呼它们)^①。我们可以直接看到,这里考虑了很多种语法模型(在特殊的模型中包括前置词和小品词)。自然,这些搭配的质量要好于机器产生的搭配,因为我们认为这些搭配是手工编辑产生的。

表 5.18 “BBI Combinatory Dictionary of English”中单词 *strength* 和 *power* 的搭配

strength	power
to build up ~	to assume ~
to find ~	emergency ~
to save ~	discretionary ~
to sap somebody's ~	~ over [several provinces]
brute ~	supernatural ~
tensile ~	to turn off the ~
the ~ to [do X]	the ~ to [do X]
[our staff was] at full ~	the balance of ~
on the ~ of [your recommendation]	fire ~

下面通过论及一些值得我们特别关注的搭配的子类,来结束关于搭配概念的讨论。

带有少许语义内容的动词,如 *make*, *take* 和 *do* 这样的动词,我们称之为搭配中的“活泼动词”(light verb),比如 *make a decision* 或者 *do a favor*。在这里很难讲 *make*, *take* 和 *do* 到底有什么含义,使我们只能使用 *make a decision* 而不用 *take a decision*,或者只能用 *do a favor* 而不能用 *make a favor*。但是由于许多计算语言学的目的,活泼动词和一个特殊名词的正确组合必须被确定。如果这种组合在机器可读的词典中没有,则需要从语料库中提取。Dras and Johnson (1996)探求了解决这个问题的方法。

动词小品词结构(Verb particle construction)或动词短语(phrasal verb)是英语词典中很重要的一部分。在英语中有很多动词,如 *to tell off* 和 *to go down*,是由一个主要动词和一个小词组成的。这些词通常相应于其他语言中的一个单独的词(如法语中的 *réprimander*, *descendre*)。这种结构是经常不带其他邻近词搭配的一个很好的例子。

专有名词(Proper nouns)(又称专名)在计算语言学的工作中也被包括在了搭配的范畴之

① 我们没有使用 *strong* 和 *powerful* 的搭配作为例子,是因为在词典中没有这两个形容词的词条。

内,虽然它和词的搭配有很大的不同。它们中的大多数和文章中以相同形式重复出现的固定短语很相近。

术语表达(Terminological expressions)或者术语短语是指技术领域中的概念和事物。虽然它们的组成结构通常是很清楚的(例如 *hydraulic oil filter*),但是识别它们仍然很重要,因为我们要确定在整个科技文本中对它们的处理方式的一致性。例如,在翻译手册时,要确保所有的 *hydraulic oil filter* 都翻译成一个相同的词。如果使用了两个不同的翻译词(即使这两个翻译词在语义上有相同的意思),那么这个翻译手册的读者都会认为我们描述了两个不同的实体,因而产生混淆。

术语搭配应用的最后一个例子牵涉到更广范围的现象,我们来关注搭配显示的不变性的不同级别。这种极端情况反映在词典的用法注释项中,描写近义词之间的微小差别的用法,如 *answer* 和 *reply* (*diplomatic answer* 对比于 *stinging reply*)。为了使生成文本更自然,这种类型的搭配的使用非常重要,但是即使在这里发生错误,也不太可能引起一个致命的错误。另一种极端的情况是完全固定的表达形式,如专有名词和成语。在这样的情况下只有一种表达形式,并且任何一个改变都会完全背离我们要表达的意思。幸运的是,对于搭配,它越不符合组合构词法并且越重要,那么它也就越容易自动获取。

5.6 深入阅读

(Stubbs 1995)深入地讨论了英国传统的经验主义语言学。

在大部分普通统计学书籍中都有关于 t 检验的论述。通常需要阅读的参考文献是 (Snedecor and Cochran 1989:53) 和 (Moore and McCabe 1989:541)。对于有较少数学背景的学生来说, Weinberg and Goldberg (1990:306) 和 Ramsey and Schafer (1997) 更容易理解。这些书籍也包括了 χ^2 检验,但是没有我们讨论的其他特殊检验。

在搭配发现方面最先公开发表的文献之一是 (Church and Hanks 1989), 后来又发表了 (Church et al. 1991)。作者对基于语料库的新型词典 (Sinclair 1995) 给予了关注, 并且开发了一个计算词典编撰程序, 这个程序结合了语料库的验证信息、计算方法和人工判断, 建立了一个更加全面的辞典, 更好地反映了语言的实际使用。

现在有很多自动处理语料数据的方法, 词典编著者可以从中获益。一个词典编著者在查看了大量的词汇使用例子之后确定了词典的条目。如果这些例子可以按照搭配和其他标准(例如, 文本的话题)进行预分类, 那么这个词典编著过程将会变得更有效率。例如, 成语动词有时会在词典中被忽略, 因为它们不是被分开的词。基于语料库的方法显然对词典编撰者来说是很重要的。此外, 一个平衡语料库将会揭示哪种用法最频繁, 因而也对于词典可能的使用者最重要。不同的测试 (t 检验) 对于编撰用法注释项和精确定义词汇不同用法之间的区别都是有用的。其中一些技术已经开始用到下一代词典的编撰中, 参见 (Fontenelle et al. 1994)。

最后, 一种新型的词典已经可以从这些工作中浮现出来, 即词典和语料库相结合的一种词典。在这样的词典中词条和语料库证据互相支持, 并且被有机地组合到一起。COBULID 词典已经具有了一些这样的特点, 参见 (Sinclair 1995)。因为对于电子词典来说空间已经不是问题了, 所以对于有兴趣的用户, 大量语料库中的例子可以整合到词典的词条中去。

我们已经论述过统计语料库分析对于单一语言词典编撰的价值,这种方法也可以应用到双语词典的编撰中,至少对于对齐的语料库是有用的,参见(Smadja et al. 1996)。

搭配的其他一些重要的应用是在信息检索(IR)中。如果用户查询和文档之间的相似性可以被常用的搭配而不是常用词来确定,检索的准确率会提高,参见(Fagan 1989; Evans et al. 1991; Strzalkowski 1995; Mitra et al. 1997)。Lewis and Jones (1996)和Krovetz (1991)深入讨论了搭配发现和自然语言处理的方法应用到信息检索中的问题。Nevill-Manning et al. (1997)讨论了一种可替代的非统计学方法使用短语在信息检索中的作用。(Steier and Belew 1993)提出了一个有趣的研究课题,当我们从一个子领域过渡到一般的非受限领域时,短语的处理(如短语的加权)将会怎样改变。例如, *invasive procedure* 是完全符合组合构词法的,并且在医学论文子领域中是一个人们不感兴趣的搭配,但是它处在一个很多领域混合的一般领域中的时候,就变成了一个令人感兴趣的非组合构词法的搭配。

搭配的其他两个重要应用分别是自然语言生成(Smadja 1993)和语言的信息检索(Hull and Grefenstette 1998)。

我们还没有涉及的一个重要领域是专有名词的识别,这些专有名词可以看成是一种搭配。由于新的人名、地名和其他实体随时可以产生和被命名,所以专有名词不能被词典完全覆盖。对专有名词自身也存在着一系列挑战:指代(coreference)(怎样才能说, *IBM* 和 *International Business Machines* 是指一个相同的实体),消歧(disambiguation)(*AMEX* 什么时候指的是 *American Exchange*, 什么时候指的是 *American Express*?),还有分类(classification)(这个文本中所指的新实体是人名、地名还是公司名?)。在这个方面的最早研究之一是(Coates-Stephens 1993)。Donald(1995)专注于词汇语义模式的研究,这些模式可以被用于专有名词的识别和分类。Mani and MacMillan(1995)和Paik et al. (1995)设计了一种按类型对专有名词进行分类的方法。

我们在这里没有论述到的,经常用来衡量搭配的重要性的一个方法是 z 积分,它和 t 检验很相近,已经应用到了几个软件包和文本分析的工作平台中(Fontenelle et al. 1994; Hawthorne 1994)。 z 积分仅仅可以用在方差已知的情况下,但是在很多统计自然语言处理的应用中并不满足这个条件。

Fisher 的严格检验(exact test)是另外一个统计检验,它可以用来判断一个观测集是否达到了我们期望的程度。与 t 检验和 χ^2 检验相反,它适合用到很小的计数情况下。但是,它很难计算,并且不太清楚实际得到的结果和 χ^2 检验是否会有很大的不同,参见(Pedersen 1996)。

还有其他一些搭配发现方法,如搜索词序列中的一些点,这些点具有低不确定性或高不确定性,这些不确定性是指在这个词的下一个(或者前一个)词将是什么。带有高不确定性的点很可能是一个短语的边界,它可以依次作为一个搭配开始或结束的候选。反之,带有低不确定性的点可能处于一个搭配的内部。(Evans and Zhai 1996)和(Shimohata et al. 1997)论述了使用这种类型的信息发现搭配和短语的两种方法。

5.7 习题

5.1 [★]在表 5.2 中加入一些搭配识别常用的词性标记模式,包括多于两个标记的模式。

5.2 [★]挑选一个有你名字出现的文档(一封电子邮件、大学成绩单或一封信)。Justeson和

Katz的过滤器可以把你的名字识别成一个搭配吗?

- 5.3 [★]上文中我们使用万维网作为辅助的语料库,因为 *strong tea* 和 *powerful tea* 都没有在纽约时报语料库中出现。修改 Justeson 和 Katz 的方法,使它可以使用万维网作为常用的资源。
- 5.4 [★★]在一个你自己选定的语料库中,使用 t 检验识别最重要的非独立二元组。
- 5.5 [★]对于二元组 *new companies*, 它的 t 值接近 1.0 是一个偶然的现象。试用一个语料库中的如下数据计算 *new companies* 的 t 值,以证明这一点。这些数据是: $C(\text{new}) = 30\,000$, $C(\text{companies}) = 9000$, $C(\text{new companies}) = 20$, 并且语料的大小 $N = 15\,000\,000$ 。
- 5.6 [★]我们可以考虑使用方差改进 5.2 节的方法。事实上, Smadja 已经这样做了, (Smadja 1993) 中描述了他的算法, 和 t 检验有一些相似性。
使用式(5.3) 对于可能的搭配计算 t 值, 并使用 5.2 节的计算方法计算均值和方差, 以代替公式中的 $\bar{x}$ 和 s^2 , 并且 (a) 假设 $\mu = 0$, (b) 假设 $\mu = \text{round}(\bar{x})$, 即四舍五入取整。注意, 我们不是在测试二元组, 而是在测试以任意一个固定小距离出现的词对是否是搭配。
- 5.7 [★★]正如前面指出的, 如果使用一个停止词表进行预过滤, 那么几乎所有的二元组的出现概率都会大大超过偶然出现的概率。请验证, 如果不过滤出这些功能词, 会有很大一部分二元组的出现概率小于偶然出现的概率。
- 5.8 [★★]对于你选择的一个语料库, 应用差异的 t 检验。测试下面的词对, 或者测试你的语料库中合适的词对的差异。这些词对是: *man/woman*, *blue/green*, *lawyer/doctor*。
- 5.9 [★]从式(5.6)推导出式(5.7)。
- 5.10 [★★]在你选择的语料库中寻找一些术语, 使它们可以最好地区别语料库的第一部分和第二部分。
- 5.11 [★★]随机选一个语料库重做上面的练习。现在你应该发现只有少数的术语是有意义的。但是毕竟还是有, 为什么? 从相同的资源中得来的语料库是不是没有差异? 在不同置信水平上做这个练习。
- 5.12 [★★]计算你选择的两个语料库的相似度。
- 5.13 [★★] Kilgariff 和 Rose 的语料库相似性度量也可以用来评价语料库的同质性。他们通过构造一系列随机分割, 把语料库划分为一对子语料库, 然后对每一个子语料库进行测试。如果大部分的测试结果是相似的, 那么这个语料库就是同质语料库。请你选择一个语料库, 使用这种方法进行测试。
- 5.14 [★★] 5.1 节中的 Justeson and Katz 的词性标记过滤器可以应用到本章的搭配发现的任何一种方法中。选择一种方法并且修改这种方法, 使之包括一个词性标记过滤器, 并说明这种修改过的方法会有什么优点?
- 5.15 [★★★]请读者选择为翻译者所接受的一种方法或者由几种方法组合而成的方法, 为翻译者设计并实现一个搭配发现的工具。

-
- 5.16 [★★★]请读者选择为词典编撰者所接受的一种方法或者由几种方法组合而成的方法,为词典编撰者设计并实现一个搭配发现的工具。
- 5.17 [★★★]现在很多新闻中都有公司名的标注。例如,所有关于General Electric Company的公司名都被标注成一个相同的标记,无论使用这个公司名的哪种变化形式。例如 GE, General Electric 或 General Electric Company。为了发现公司名,设计并实现一个搭配发现工具。考虑怎样才能使发现公司名变体的过程部分自动化。

第6章 统计推理:稀疏数据集上的 n 元语法模型

统计自然语言处理的目的是针对自然语言领域进行统计推理。通常,统计推理(Statistical inference)包括处理数据(具有某些未知的概率分布)和根据这些数据分布得到一些推论。例如,我们可以观察语料库中介词短语附着关系的大量实例,并且据此预测英语中介词短语的附着关系。在本章的讨论中,我们把这个问题分成三个部分进行阐述(虽然它们在相当程度上是互相交叉的):把训练数据划分成等价类,为每个等价类寻找一个好的统计估计,对多个估计进行合并。

作为一个常用的统计估计的例子,我们将考察经典的语言建模问题,即根据当前词预测下一个词。它是语音识别或光学字符识别的基础,同时也被用来进行拼写纠错、手写体识别和统计机器翻译。这类任务经常被称为 Shannon 游戏(Shannon 1951):在一篇文本中猜测下一个字母。这个问题已经经过了很好的研究,并且很多估计方法最初就是针对这个任务被开发出来的。但是一般而言,我们在本章中提及的方法并不是特别针对这个任务的,它们也可以被用到其他的语言学任务中,如语义消歧和概率句法分析。词汇预测任务是这项技术可以解决的一个简单明了的问题。

6.1 Bins:构造等价类

6.1.1 可靠性和辨别性

通常,为了对一个特征进行推导,我们希望找到模型中可以预测它的其他特征。在这里,假设过去的行为可以很好地预测将来要发生的事情(也就是说,这个模型大体上是稳定的)。这样就有了分类任务:我们试图基于各种类别特征(classificatory feature)预测目标特征(target feature)。在分类的时候,我们把数据有效地划分到具有不同类别特征的等价类别中去,并且使用这些等价类帮助我们在新的数据中预测目标特征的值。这就意味着我们在分类的时候做了独立性假设(independence assumption):每一个类别特征中的数据或者与其他特征无关,或者与其他特征的相关性很弱,以至于我们忽略它也不会带来多大的损失。我们使用的类别特征越多,在确定目标特征的未知概率分布时的条件就会越多。换句话说,把数据划分到很多等价类中,可以使我们有更好的辨别性。但另一方面,如果使用了太多的等价类,那么对于每个类别来说,它可能不包含或者包含很少训练数据,这样就无法对这个类别的目标特征做出可信的统计估计。权衡这两个标准,找到一个合适的类别数,是我们的首要目的。

6.1.2 n 元语法模型

单词预测任务可以看做是估计下面的概率方程 P :

$$P(w_n | w_1, \dots, w_{n-1}) \quad (6.1)$$

在这样一个随机问题中,我们使用了先前词,也就是用历史来预测下一个将要出现的词。通过观测大量的文本,我们会知道什么词趋向于出现在另外一些词的后面。

对于这个任务,我们不可能单独考虑每个文本历史。在大多数情况下,我们遇到的句子是以前从没有见到的,没有与之一样的可以预测它的文本历史,比如我们可能见到过这个句子的开始,但是它在这一次却有不同结尾。所以我们需要一个可以把文本历史组合到一起的方法,对于预期将要出现的那个词,在某种程度上可以给出一个合理的预测。组合文本历史的一种可能方法是马尔可夫假设,这个假设认为只有当前的上下文(前面的很少几个词)可以影响下面要出现的词。如果我们构造一个模型,这个模型中的所有历史都是前 $n-1$ 个已经出现的词,那么我们就有了一个 $(n-1)$ 阶马尔可夫模型,或者称为 n 元语法模型(这个 n 元语法模型的第 n 个词就是我们要预测的词)。

在继续建模工作之前,我们简短地讨论一下命名问题。人们经常使用的 n 元语法模型是 $n=2, 3, 4$ 的情况,在这些情况下它们分别被称为 bigram(二元语法), trigram(三元语法)和 four-gram(四元语法)模型。任何一个阅读本书的经典主义者可能都会使用工程规范来质疑上面的命名:gram 是一个希腊词根,所以我们应该使用希腊字母作为前缀。事实上,Shannon 使用的术语是 digram,但是在最近几十年的教学中这个用法已经不存在了。而且,作为非说明性语言,我们认为在大学中使用的英语、希腊语和拉丁语的奇怪组合是非常有趣的。所以我们不会试图摒弃这种用法^①。

原则上,我们希望 n 元语法模型的 n 越大越好,比如像下面的词串:

Sue swallowed the large green ____ (6.2)

swallowed 对 green 后面将要出现的词还有很强的影响, pill 或者 frog 是可能的后续词,但是 tree, car 和 mountain 就不太可能,尽管它们出现在 the large green 之后通常很自然。另外一个问题是,如果我们把数据划分到太多的类别中,那么就有大量的参数要去估计。例如,保守估计一个说话者使用的词表有 20 000 个词,那么我们的参数数量的估计值如表 6.1^② 所示。

表 6.1 n 元语法模型参数数量

模型	参数
1st order (二元语法模型)	$20\ 000 \times 19\ 999 = 400\ \text{million}$
2nd order (三元语法模型)	$20\ 000^2 \times 19\ 999 = 8\ \text{trillion}$
3th order (四元语法模型)	$20\ 000^3 \times 19\ 999 = 1.6 \times 10^{17}$

我们可以看到,即使有一个很大的语料库,编程实现一个五元语法模型并不是一个很现实的想法。由于这个原因,当前通常使用的 n 元语法模型系统是 bigram 和 trigram(并且通常使用的是一个比较小的词表)。

减少参数数量的一个方法是减小 n 的值,但是 n 元语法模型并不是构成历史等价类的惟一方法。在其他一些分类的操作中,我们可以考虑使用“词干化”(stemming)(删除词语的变形结尾)或者语义类的方法(通过使用已经存在的辞典,或者引用一些聚类的方法)来构造等价

① 有些人试图使用 quadgram 来代替 four-gram,但是这并不是拉丁语数字前缀的正确用法(正确的用法应该是 quadri-gram,类似于 quadrilateral),正确的希腊语数字前缀的用法是“a tetragram model”。

② 给定一个确定的模型空间(在这里是词的 n 元语法模型),它的参数就是我们在模型空间中确定一个特殊的模型时需要的参数。

类。这些方法有效地减少了我们在 n 元语法模型中使用词表的规模。而且,我们完全没有必要使用 n 元语法。实际上有很多方法可以构造历史等价类,只是这些方法都比 n 元语法更复杂一些。上面的例子已经指出,在一个句子中谓语的知识是很有用的,所以我们可以设想一个模型,它基于先前出现的词和前面的谓语(无论它离得多远)来预测下一个词。但这个模型很难实现,因为我们首先要有一个非常准确的方法来识别句子的主要谓语。因此,在本章中我们仅仅使用 n 元语法模型,其他的技术在第 12 章和第 14 章中会涉及到。

对于任何一个有语言学背景的人来说,如果一个语言结构模型只根据前面出现的词来预测下一个词,而不涉及到句子的结构,这样的想法似乎是很荒谬的。但事实上,邻近上下文中出现的词汇同现、语义和基本的句法结构关系都是下一个词的很好的预测器,而且这样的系统也取得了很好的结果。实际上,在预测下一个词的这种纯粹的线性任务中,很少有模型能与 trigram 模型相匹敌。

6.1.3 构建 n 元语法模型

在本章的结束部分,我们将会构建一些实际的模型并且给出一些实验结果。通过使用本书配套网站上提供的工具和数据,读者可以重现我们的结果。我们将使用的文本是 Jane Austen 的小说,它也可以在本书配套网站上得到。这个语料库有两个优点:(i) 由于 Gutenberg 项目的工作,它可以免费使用,(ii) 它的规模不大。当然,在很多情况下小的语料库也是一个缺点。正如前面讨论的,由于 n 元语法模型的参数数量非常多,所以在足够大的数据集上进行训练时, n 元语法模型才能更好地工作。但是,这样的训练需要大量的 CPU 时间和硬盘空间,因此对于教材所用的例子来说,一个小的语料库更合适。即使如此,试图重现我们的例子也需要有 40 MB 的硬盘空间。

通常,构建模型的第一步是处理语料。Gutenberg 项目的 Austen 文本是非常清晰的无格式 ASCII 文件。但是标点符号的标记是一个问题,它比用空格分割词语处理起来更复杂一些。可以使用一种非常简单的搜索-替换的模式来解决这个问题,它删除所有标点符号并用空格替换来分割词语(细节问题参见本书配套网站)。我们使用 *Emma*, *Mansfield Park*, *Northanger Abbey*, *Pride and Prejudice* 和 *Sense and Sensibility* 作为我们建立模型的语料,保留 *Persuasion* 作为测试语料。这个训练语料库(比较小)的词数为 $N = 617\,091$,包含一个 14 585 个词的词表 V 。

通过上面的方法简单删除标点符号后,我们的文件变成了一个长的词序列。这些工作实际上不会浪费太多的时间。根据一般的感受,虽然每个句子都趋向于使用特殊的开头,但是句子之间没有太强的联系。所以,人们在文本中标注句子时,通常都是用 SGML 标记 $\langle s \rangle$ 和 $\langle /s \rangle$ 。而且句子开头的概率计算也不依赖前一个句子的结尾词,而是依赖一个称为“句子开头”的上下文。此外还要注意的,我们没有去掉某些情况的差别,所以在数据中那些大写字母开头的单词会被保留下来,以此来近似地指定句子开头。

6.2 统计估计

给定一定数量的训练数据,并把它们归为某些特征类别之后,接下来的一个目的是,对这些数据的目标特征找出一个好的概率估计。对于 n 元语法模型的例子,我们感兴趣的是概率 $P(w_1 \cdots w_n)$ 和预测任务 $P(w_n | w_1 \cdots w_{n-1})$,因为:

$$P(w_n | w_1 \cdots w_{n-1}) = \frac{P(w_1 \cdots w_n)}{P(w_1 \cdots w_{n-1})} \quad (6.3)$$

估计条件概率分布可以简化为估计 n -gram 的未知概率分布^①。

假设训练文本包含 N 个词。如果在文本开头加入 $n-1$ 个虚拟的开始符号,就可以说这个语料库是由 N 个 n -gram 组成的,并且在所有情况下预测下一个词的条件(可用的词数)都是一致的。设 B 是类别数(等价类)。设 V 是词表的大小,对于预测下一个词的任务,我们有域 V ,对于估计不同的 n -gram 概率的任务,我们有域 V^n 。设 $C(w_1 \cdots w_n)$ 是训练文本中某一个 n 元组出现的频率,在训练文本中出现 r 次的 n -gram 的个数是 N_r (例如, $N_r = |\{w_1 \cdots w_n : C(w_1 \cdots w_n) = r\}|$)。这个频率的计数在后面设计的概率估计方法中经常用到。这些符号在表 6.2 中进行了总结。

表 6.2 统计估计中的符号

N	训练实例的数量
B	训练实例的类别数
$w_{1:n}$	训练文本中的一个 n 元组 $w_1 \cdots w_n$
$C(w_1 \cdots w_n)$	训练文本中的一个 n 元组 $w_1 \cdots w_n$ 出现的频率
r	一个 n 元组的频率
$f(\cdot)$	一个模型的频率估计
N_r	含有 r 个训练实例的类别数目
T_r	在更多数据中频率为 r 的 n 元组的总数
h	先前词的历史

6.2.1 最大似然估计

相对频率的最大似然估计

无论怎样形成等价类,都会得到包含一定数量训练实例的分类。假设一个 trigram 模型,它用上下文中前两个词来预测下一个词,并且重点关注两个先前出现的词是 *comes across* 的情况。在一个确定的语料库中,作者找到了词串 *comes across* 的 10 个训练实例,在它们中有 8 次后面接 *as*, 1 次接 *more*, 1 次接 *a*。这种情况下如果要估计下一个词,应该使用哪种概率估计呢?

最明显的答案(至少从通常的观点来看)是使用相对频率(relative frequency)作为概率估计。

$$\begin{aligned} P(as) &= 0.8 \\ P(more) &= 0.1 \\ P(a) &= 0.1 \\ P(x) &= 0.0 \quad \text{当 } x \text{ 不是前 3 个词当中任意一个时} \end{aligned}$$

^① 但是,在平滑的时候可以选择对 n -gram 概率估计进行平滑,也可以直接对条件概率分布估计进行平滑。对于很多方法,这样做的结果是不同的,因为在后一种情况下,要分别平滑大量的条件概率分布(这些概率分布通常要通过某种方法进行聚类)。

这种估计被称做最大似然估计(MLE):

$$P_{\text{MLE}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n)}{N} \quad (6.4)$$

$$P_{\text{MLE}}(w_n | w_1 \cdots w_{n-1}) = \frac{C(w_1 \cdots w_n)}{C(w_1 \cdots w_{n-1})} \quad (6.5)$$

如果固定观测数据,并且认为所有的参数空间服从某一分布(在这里是 trigram 模型),统计学家就把这个公式称为似然函数。之所以称为最大似然估计,是因为它选择的参数值对于训练数据给出了最高的概率^①。这个估计正如上面的公式所示。它没有在训练语料中未出现的事件上浪费任何概率,而是使观测到的事件的概率达到最高(也就是说,服从正常随机约束情况下的最高值)。

但是一般来讲,MLE 并不适合于自然语言处理中的统计推理。这是由于数据稀疏引起的(即使当我们使用一个大的语料库时,也会有数据稀疏问题)。一般常用词并不多,词汇的大部分都是不常用的,因此包含它们的 n -gram 也是很稀少的。MLE 对于没有找到的事件指定了一个 0 概率,同时因为一个长串的概率通常是由它的子串概率的乘积得到的,所以这个 0 值就会进行传导。那么,对于一个句子,当它的一个 n 元组正好没有在训练语料中出现时,我们就会得到一个很差的概率估计(0 概率)^②。对于上面举的那个例子,最大似然估计不会认为还有其他词可以接在 *comes across* 的后面,例如 *the* 和 *some*。

下面给出一个数据稀疏的例子。(Bahl et al. 1983)从 IBM 的 Laser Patent Text 语料库中得到了一个 1 500 000 个词的训练语料,在这个语料上进行训练之后,使用从相同语料库得到的测试文本进行测试,结果是测试语料中共有 23% 的 trigram 是训练中没有看到的。上述训练语料库相对于现在的标准来说是很小的,所以有人可能希望通过收集更多的语料来解决数据稀疏的问题。虽然这种想法初看起来是有希望的(如果我们收集了 100 个关于 *comes across* 的例子,那么可能会找到 *comes across* 后面接 *the* 和 *some* 的例子),但实际上这样做并不能完全解决问题。虽然语言中频繁出现的事件是有限的,但是越来越稀有的事件的概率分布永远不会完结,所以不可能搜集到足够的数据来得到稀有事件的完整概率分布^③。对于 *comes across* 的例子来说,它可以接任何一个数字,但是我们不可能收集到每个数字的实例。一般来说,需要一种更好的估计方法,它给那些在训练数据中没有出现的事件一定大小的概率。

通过稍微减少已观察到的事件概率的大小,同时把少量概率分配到没有看到过的事件上,那么上述方法就可以有效地工作。这通常被称为折扣法(discounting method)。折扣过程通常称为平滑(smoothing),这大概是因为一个没有 0 值的分布比有 0 值的分布更平滑。在下面的章节中我们将会考察很多平滑方法。

① 即,某一个 n 元组的出现被假设成一个服从二项分布的随机变量(也就是说,每一个 n 元组是独立于文本的)。这是一个非常不正确(虽然是可用的)的假设。第一,每个 n 元组互相重叠并且因此可以部分地确定下一个 n 元组;第二,实词具有一定的局部性(如果你在一篇文章中使用了一个词,那么很可能再次使用它),正如我们在 15.3 节中讨论的那样。

② 换句话说,如果我们的概率模型指定 0 概率给任何一个实际上存在的事件,那么相对于真实概率分布的交叉熵和相对熵就会是无限大。也就是说,我们试图模型化的概率函数的结果是最坏的。

③ 参照 Zipf 规则,一个观测到的现象,在词语频率和它的频率的排列位置之间存在一个近似的倒数关系。我们在 1.4.3 节中已经对其进行了讨论。

使用 MLE 估计 Austen 的 n 元语法模型

以 Austen 语料库为基础,建立 n 有不同取值的 n 元语法模型。编程实现这个目标非常简单,通过计算 n 元组和 $n-1$ 元组的频率,然后相除就可以得到 MLE 概率估计,而且在本书配套网站上也有这样的软件。

在实际的系统中,对所有词计算 n 元组的概率是不现实的。通常只是计算最常见的 k 个词的 n 元组,而其他的词被认为是未登录词(Out-Of-Vocabulary, OOV),并且标上一个单一的符号,例如 $\langle \text{UNK} \rangle$ 。一般来说,对于只在训练语料中出现一次的词(hapax legomena),这种方法是可行的。在一些领域中稀有的数字具有明显的语义和分布相似性,因此可以有两个未登录词标记,一个用于标记数字,另一个用于标记其他的词。根据 Zipf 的词汇分布规律,截断低频率词将会大大减少参数空间(并且也会减少系统构建时的内存需求),但是会对系统的质量有所影响。hapax legomena 通常占词形(type)统计的一半,而只占词次(token)统计的一小部分。

对于测试语料库persuasion中的部分句子,我们使用从训练语料中计算出来的条件概率来计算这些句子中下一个词的概率。在后面会更详细地讨论测试语料的问题,在不同数据上评估模型是很重要的,否则我们无法得到关于这个模型预测语言模式的性能的一个公正评价。表 6.3 中给出了一些概率分布的测试结果(黑体表示真实的下一个词)。unigram (1-gram)分布完全忽略了上下文的信息,只是简单地使用了不同词出现的频率。这样的方法并不是完全没有用的,因为大部分句子中的大部分词都是经常出现的词汇。bigram 模型使用前一个已经出现的词来帮助预测下一个词。一般来说,bigram 提供的信息已经很多了,和 unigram 相比,它给出了一个更好的模型。在有些情况中,下一个词的估计概率会提高大约一个数量级(如 *was*, *to*, *sisters*)。然而,我们要注意bigram并不保证一定增加概率估计。对于 *she* 的概率估计事实上就下降了,因为 *she* 在 Austen 的小说(主要是关于女性的书籍)中非常常用,但是 *she* 一般不会出现在名词*person*后面,虽然当前面是一个副词短语时它也很有可能,例如 *in person*。预测在 *was* 之后出现*inferior*的失败表明,数据稀疏的问题已经出现了。

当 trigram 模型可以工作的时候,它能得到非常好的结果。例如,对于 *was* 跟在 *person she* 后面的情况,它给出了一个 0.5 的概率估计。但是,通常它是不适用的,这是由于前面的 bigram 在训练语料中没有看到,那么就没有了先前词的概率分布,或者是由于只有很少的词跟随在这个 bigram 后面,数据非常稀少以至于结果估计非常不真实。例如,二元组 *to both* 在训练语料中共出现 9 次,两次接 *to*,其他 7 次都是接不同的词,这些词中有几个已在表 6.3 中列出。在这样的数据密度上不可能正确地建立一个概率模型。four-gram 更是完全不能使用。four-gram 模型在一般情况下是不适用的,除非可以在一个几千万词的数据集上进行训练。

考察表中的数据可以得出这样的策略:对于一些有足够多训练数据的词串,使用高阶的 n 元语法模型,当训练数据不充分的时候,回退到低阶的 n 元语法模型。这是一个广泛使用的策略,我们将在 6.3 节中讨论这种策略,但是它本身并不是 n -gram 估计问题的一个完备的解决方案。例如,训练语料中有非常多的词跟随在 *was* 的后面,1481 个词形,共计 9409 个词次,但是 *inferior* 却不在其中。与此相似,虽然训练语料中有大量实例,但还是有很多词没有出现,包括非常简单的词,如 *decides* 或 *wart*。所以无论怎样混合评估,我们仍然需要一种方法,给那些在训练语料库中没有出现的词或者 n 元组一个非零概率,因此我们首先要解决这个问题。

表 6.3 Persuasion 语料库中的一个句子中每个后继词的概率。表中使用最大似然估计的 n 元语法模型(对于不同的 n 值)计算了下一个词的概率分布。不同词语的预测似然值大小排序显示在第一列。原句中的下一个词用斜体显示在表的顶端,并且在表中用粗体表示

<i>In</i> <i>person</i>	<i>she</i>	<i>was</i>	<i>inferior</i>	<i>to</i>	<i>both</i>	<i>sisters</i>
1-gram	$P(\cdot)$	$P(\cdot)$	$P(\cdot)$	$P(\cdot)$	$P(\cdot)$	$P(\cdot)$
1	the	0.034	the	0.034	the	0.034
2	to	0.032	to	0.032	to	0.032
3	and	0.030	and	0.030	and	0.030
4	of	0.029	of	0.029	of	0.029
...						
8	was	0.015	was	0.015	was	0.015
...						
13	she	0.011	she	0.011	she	0.011
...						
254			both	0.0005	both	0.0005
...						
435			sisters	0.0003		sisters
...						
1701			inferior	0.000 05		
2-gram	$P(\cdot person)$	$P(\cdot she)$	$P(\cdot was)$	$P(\cdot inferior)$	$P(\cdot to)$	$P(\cdot both)$
1	and	0.099	had	0.141	not	0.065
2	who	0.099	was	0.122	a	0.052
3	to	0.076	the	0.033	her	0.048
4	in	0.045	to	0.031	have	0.027
...						
23	she	0.009			Mrs	0.006
...						
41					what	0.004
...						
293					both	0.0004
...						
∞			inferior	0		
3-gram	$P(\cdot In, person)$	$P(\cdot person, she)$	$P(\cdot she, was)$	$P(\cdot was, inf.)$	$P(\cdot inferior, to)$	$P(\cdot to, both)$
1	UNSEEN	did	0.5	not	0.057	UNSEEN
2		was	0.5	very	0.038	the
3				in	0.030	Maria
4				to	0.026	cherries
...						her
∞			inferior	0	both	0
4-gram	$P(\cdot u, l, p)$	$P(\cdot l, p, s)$	$P(\cdot p, s, w)$	$P(\cdot s, w, i)$	$P(\cdot w, l, t)$	$P(\cdot i, t, b)$
1	UNSEEN	UNSEEN	in	1.0	UNSEEN	UNSEEN
...						
∞			inferior	0		

6.2.2 Laplace 法则、Lidstone 法则和 Jeffreys-Perks 法则

Laplace 法则

最大似然估计的缺点强迫我们去考察更好的概率估计。最古老的解决方法是使用 Laplace 法则(1814; 1995)。依照这个法则,

$$P_{\text{Lap}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n) + 1}{N + B} \quad (6.6)$$

这种处理方法常常被非正式地称为加 1 法,它把一小部分概率有效地转移到了未知事件上。

这样做不是简单无道理的移动概率,如果我们对事件假设了一个统一的先验概率(也就是说,每个 n 元组都有相同的可能性),那么事实上它就是贝叶斯估计。

然而,我们要注意到,Laplace 法则给出的估计方法依赖于词表的大小。对于一个大词表上的稀疏数据集,Laplace 法则实际上把太多的概率转移到了未知事件上。

考虑 Church and Gale(1991a)讨论的数据,他们对 bigram 的各种估计方法都进行了讨论。他们的语料库是一个 44 000 000 个词的美联社(AP)新闻语料库,并由此生成了一个 400 653 个词的词表(保留了格差别,以连字符分割)。注意这个词表的大小,它意味着有 1.6×10^{11} 个可能的 bigram,所以这个空间中的任意一个 bigram 先验零概率都有可能出现在语料中。也就是说,在式(6.6)的计算中, B 要远远大于 N ,所以在这样的情况下 Laplace 规则是不能令人满意的。Church 和 Gale 使用了一半的语料(22 000 000 个词)作为训练语料。表 6.4 中显示了他们讨论的各种方法得到的期望频率估计,并且计算了 Laplace 法则的频率估计。概率估计等于频率估计除以 n -gram 数目, $N = 22\,000\,000$ 。对于 Laplace 法则,一个出现 r 次的 n -gram 的概率估计是 $(r+1)/(N+B)$,所以它的频率估计就是 $f_{\text{Lap}} = (r+1)N/(N+B)$ 。频率估计对人们来说更加容易解释,因为更容易看到折扣的效果。

表 6.4^① Church and Gale(1991a)的 AP 数据的频率估计。前 5 列显示了对于不同的估计方法,在训练语料中出现 r 次的 bigram 的频率估计值。 r 表示最大似然估计, $f_{\text{empirical}}$ 是测试集中的确认结果, f_{Lap} 是加 1 法, f_{del} 是删除插值法, f_{GT} 是 Good-Turing 估计法。最后两列给出了频率的频率和某一频率的 bigram 在新文本中出现的次数

$r = f_{\text{MLE}}$	$f_{\text{empirical}}$	f_{Lap}	f_{del}	f_{GT}	N_r	T_r
0	0.000 027	0.000 137	0.000 037	0.000 027	74 671 100 000	2 019 187
1	0.448	0.000 274	0.396	0.446	2 018 046	903 206
2	1.25	0.000 411	1.24	1.26	449 721	564 153
3	2.24	0.000 548	2.23	2.24	188 933	424 015
4	3.23	0.000 685	3.22	3.24	105 668	341 099
5	4.21	0.000 822	4.22	4.22	68 379	287 776
6	5.23	0.000 959	5.20	5.19	48 190	251 951
7	6.21	0.001 09	6.21	6.21	35 709	221 693
8	7.21	0.001 23	7.18	7.24	27 710	199 779
9	8.26	0.001 37	8.18	8.25	22 280	183 971

尽管每个没有在训练语料库中看到的 bigram 都被给了一个非常小的概率,但是因为这些 bigram 太多了,所以实际上有 46.5% 的概率空间都给了这些没有出现的 bigram^②。这个比例太高了,而且这样做的代价是极大地减少了频繁出现事件的估计概率。我们怎么知道这个比例太高了呢?在表中第二列中显示了一个经验的确认估计(下面将会讨论),它反映了未知的 n -gram 在更多文本中出现的可能性,可以看到未知 n -gram 的出现频率比 Laplace 规则预测得低很多,但是已知 n -gram 的出现频率比预测得要高很多^③。而且经验模型发现,在进一步的文本中只有 9.2% 的 bigram 是未知的。

① 按照本书配套网站上提供的勘误信息,整个语料库有 400 653 个词形,训练语料库有 273 266 个词形。 B 应该是 273 266,而表 4.6 的计算中却用了 $B = 400\,653$,因此计算值出现了错误。表 6.4 中的 f_{Lap} 列的第一行(0)的数字 0.000 137 应该改成 0.000 295。其他的数值计算结果,有兴趣的读者可以自己校对。——译者注

② 这个比例是这样计算的: $N_0 \times P_{\text{Lap}}(\cdot) = 74\,671\,100\,000 \times 0.000\,137/22\,000\,000 = 0.465$ 。

③ 表中的天文数字比较难处理。习题 6.2 中的一个较小的例子同样证明了这个观点。

Lidstone 法则和 Jeffreys-Perkes 法则

由于上面的过估计,统计实践中通常的解决多项式估计问题的方法是连续的 Lidstone 法则,不是加 1,而是加一个正值(通常较小) λ :

$$P_{\text{Lid}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n) + \lambda}{N + B\lambda} \quad (6.7)$$

这种方法是由保险精算师 Hardy 和 Lidstone 发明的,Johnson 证明了它可以看做是在 MLE 估计和统一的先验概率之间的线性插值。设 $\mu = N/(N + B\lambda)$,我们就可以看到:

$$P_{\text{Lid}}(w_1 \cdots w_n) = \mu \frac{C(w_1 \cdots w_n)}{N} + (1 - \mu) \frac{1}{B} \quad (6.8)$$

最常使用的 λ 值是 1/2。这个选择在理论上可以被证明是 MLE 最大化的期望,所以它有自己的名字,Jeffreys-Perks 法则,或者称为期望似然估计(Expected Likelihood Estimation, ELE)(Box and Tiao 1973:34 ~ 36)。

这种方法在实际中经常是很有用的。例如,我们可以通过选择一个小的 λ 避开如下的缺点:太多的概率空间被转移到未知事件上。但是这种方法还残留着两个缺点:(i) 需要预先猜测一个合适的 λ 值,(ii) 使用 Lidstone 法则的折扣总是在 MLE 频率上给出一个线性的概率估计,但是这和低频情况下的经验分布不能很好地吻合。

在 Austen 语料库中应用这些方法

虽然这些方法还存在着一些内在的问题,但是我们仍然试着在 Austen 语料库中应用它们,特别是 ELE。回想一下,到现在为止,对于测试语料中的句子 *she was inferior to both sisters*,仅有的概率估计是 unigram 估计,它给出了这个句子的概率(通过将表 6.3 顶部粗体显示的概率相乘获得)是 3.96×10^{-17} 。表中的其他模型,由于数据稀疏的原因,概率估计是 0 或者没有定义。

对于这个句子,现在使用 bigram 模型和 ELE 来计算它的概率估计。训练语料中 *was* 共出现 9409 次,*not* 在其后面出现了 608 次,这个训练语料中共有 14 585 个词形。所以对于条件概率 $P(\text{not} | \text{was})$,新的估计是 $(608 + 0.5)/(9409 + 14\ 589 \times 0.5) = 0.036$ 。这个估计已经被折扣过了(几乎一半都被折扣了!)。如果我们对其他词也进行相似的计算,这些结果显示在表 6.5 中的最后一列。最可能在 *was* 后出现的词的顺序没有改变,但是这些在训练语料中出现的词的概率估计已经被折扣了。没有出现在训练语料中的,事实上的下一个词 *inferior*,已经被指定了一个非零出现概率。继续用这种方法计算其他的 bigram 的概率估计,会发现这个语言模型对例句给出的概率估计是 6.89×10^{-20} 。遗憾的是,这个概率估计值实际上要比基于 unigram 计数的 MLE 概率估计值低——这反映了在 ELE 模型的结构中,所有已知事件的 MLE 概率估计都被极大地折扣了。这个结果证明了在(Gale and Church 1990a, b)标题中的一句口号:差的上下文估计还不如没有(*poor estimates of context are worse than none*)。当然,这并不是意味着所构建的模型是完全没用的。虽然它给出的概率估计非常低,但是我们仍然可以使用这些概率估计来对可能的选择进行排序。例如,模型可以正确地告诉我们,在英语中 *she was inferior to both sisters* 比 *inferior to was both she sisters* 出现的可能性要大得多,但是 unigram 估计给这两个句子的概率是一样的。

表 6.5 紧跟在单词 *was* 之后出现的单词的期望似然估计值

Rank	Word	MLE	ELF
1	not	0.065	0.036
2	a	0.052	0.030
3	the	0.033	0.019
4	to	0.031	0.017
...			
=1482	inferior	0	0.000 03

6.2.3 留存估计

怎么才能知道把概率空间的 46.5% 转移到未知事件上是太多了呢? 一种测试方法就是经验方法。我们在更多的文本中(假设来自于相同的资源)查看, 在训练语料中出现 r 次的 bigram 在更多的文本中出现的次数。这个想法的实现就是(Jelinek and Mercer 1985)的留存估计法。

留存估计法

对于每个 n -gram, $w_1 \dots w_n$, 有:

$C_1(w_1 \dots w_n)$ = 训练数据中 $w_1 \dots w_n$ 的频率

$C_2(w_1 \dots w_n)$ = 留存数据中 $w_1 \dots w_n$ 的频率

N_r 是出现 r 次(在训练文本中)的 n -gram 的数量。现在有:

$$T_r = \sum_{\{w_1 \dots w_n: C_1(w_1 \dots w_n)=r\}} C_2(w_1 \dots w_n) \quad (6.9)$$

也就是说, T_r 是所有在训练文本中出现 r 次的 n -gram 在留存数据中出现的总数量。那么这些 n -gram 的平均频率是 T_r/N_r , 所以对于这些 n -gram 之一的概率估计是:

$$P_{ho}(w_1 \dots w_n) = \frac{T_r}{N_r T} \quad \text{其中 } C_1(w_1 \dots w_n) = r \quad (6.10)$$

T 为留存数据中的词次数量。

开发和测试模型的数据集

统计自然语言处理中的一个最大的错误就是在训练数据上进行测试。这是为什么呢? 测试的目的就是去评价一个特殊模型的工作效果。所以公正测试只能在一个以前没有看过的数据集上进行。通常, 从一个数据样本得来的模型会有训练过度(overtrained)的趋向, 也就是说, 期望未来的事件和已经训练过的数据一样, 没有为其他的可能性留出足够的空间。例如, 股票市场模型有时就会遭受这样的失败。所以, 在不同的数据集上进行测试是必要的。交叉熵的计算(见 2.2.6 节)便是一个很好的例子。为了计算交叉熵, 我们使用一个大的文本样本, 按照模型计算这个文本中每个词的熵。这个计算不仅给出了模型质量的评价, 而且也给出了常用文本语言熵的上界。但是, 只有当测试数据和训练数据完全独立, 并且测试数据大到足以指示语言复杂度的时候, 这些结论才成立。如果我们在训练数据上测试, 交叉熵要比真实文本的熵低。在很多哗众取宠的情况下, 我们可以构建一个模型, 记住所有的训练文本, 并且可以预测下一个词出现的概率为 1。即使不这样做, 如果在训练数据上测试, 也可以发现 MLE 是一个出

色的语言模型,但这是不正确的结果。

所以,当使用一些数据开始工作时,应该马上把这些数据划分为训练数据和测试数据。测试数据一般只占总数据的一小部分(5%~10%),但是它必须充分保证测试结果的真实性。应该总是把注意力集中在训练数据上——使用模式识别能力去发现构造模型的线索。不要把注意力集中在测试数据上。

然而,通常由于不同的原因,需要把训练数据和测试数据再分别划分为两个部分。很多统计自然语言处理的方法,像 n -gram 的留存估计,就是从一个大规模的训练数据上得到统计量,然后在更多留存数据(held out data)或验证数据(validation data)的基础上平滑这些统计量,或者估计模型的某些参数。留存数据需要与训练数据和测试数据保持独立。在正常情况下,使用留存数据得到的参数估计要比从主要训练语料中得到的参数估计少得多,所以对于留存数据来说,它的大小比主要的训练语料小得多(通常大约是训练语料的 10%)。但是如果要准确估计模型的任何一个附加参数值,或者一些重要的缺失性能,数据量也要足够大,正如(Chen and Goodman(1996:317)中显示的那样。

统计自然语言处理研究的典型过程是,设计算法,进行训练并测试算法,发现错误,然后改正错误并重复这个过程(通常要很多次!)。但是,如果重复的次数很多,那么测试数据也被使用了很多次,而且我们根据测试数据上的效果试验了不同的算法。这意味着,多次测试模型又一次导致了训练过度。正确的方法是使用两个测试集:开发(过程)测试集(development test set)用来在它上面试验各种方法;最终测试集(final test set)用来产生最后的结果,公布算法的效果。我们应该希望看到这样的结果:在最终测试集上的效果要比在开发(过程)测试集上的效果略差(虽然有时比较幸运)。

迄今为止的讨论还没有涉及怎样选择一部分数据作为测试数据。事实上对这个问题的看法分为两派。一派认为从全部数据中随机选择一些(句子或 n -gram)作为测试数据,其他剩余部分作为训练数据。这样做的优点是,测试数据尽可能类似于训练数据(在流派、术语、作者和词表等方面)。也就是说,我们的训练样本尽可能准确地来源于测试数据的语言类型。另一种可能的方法是,保留一个大块内容作为测试数据。这样做的优点是相反的,实际上我们最终会在一个数据集上使用一个自然语言处理系统,这个数据集和训练数据一定会有些许不同,因为随时间的推移语言在话题和结构上会有变化。因此,一些人认为这种方法可以最佳地模拟由于选择测试数据带来的不同,这些测试数据相对于训练数据可能是不固定的。无论如何,如果我们选择参数的留存估计法,最好选择和测试数据相同的策略来保留数据作为留存数据,因为这样做可以使留存数据最好地模拟测试数据。这样选择也是系统结果很难比较的原因之一:在其他条件都一样的情况下,使用第二种方法的评价结果应该要稍微差一些。

在讨论测试的时候,我们还要注意其他问题。在早期的工作中,通常只是在测试数据上运行系统并给出单一的性能(混乱度、正确率或者其他什么)。但这不是一个非常好的测试方法,因为它没有给出反映系统性能的方差评价。更好的方法应该是把测试数据划分到一些相似的样本集中,比如 20 个样本集,然后在每个样本集上得出测试结果。利用这些结果,可以像以前一样得到均值性能,而且还可以计算方差,反映系统性能的波动。如果这种方法和连续的训练数据块一起使用,最好从训练数据的不同地方得到小一些的测试数据,因为这些测试数据可以很好地反映训练数据集中某一部分的情况,因此最好使用从语料库的不同部分得到的测试数据。

如果我们使用这种方法,那么系统的评价平均分要比纯粹的偶然选取测试数据的系统平

均分高,特别是当系统内部方差值较高的时候。所以对于系统比较而言,仅仅比较评价平均分是不够的。相应地,我们需要使用一种考虑均值和方差的统计测试。只有当统计测试摒弃了偶然差异的可能性之后,我们才能说一个系统好于另一个系统^①。

使用 t 检验(我们在 5.3.1 节中介绍过)比较两个系统性能的例子显示在表 6.6^②中,改编自 (Snedecor and Cochran 1989:92)。注意,在这里我们使用了样本方差 s^2 的合并估计(pooled estimate),假设两个系统的方差是相同的(在这里这个假设似乎是合理的:1375.4 和 1228.8 足够接近)。查询附录 A 中的 t 分布,我们发现,可以在置信度 $\alpha = 0.05$ 的级别,临界值 $t = 1.725$ (使用 one-tailed 测试,自由度 20)时拒绝系统 1 好于系统 2 的假设。因为我们的 $t = 1.56 < 1.725$,所以置信测试失败。尽管两个系统的平均值有很大差别,但是我们在这里也不能得出系统 1 比较优越的结论,因为它的方差很大。

表 6.6 使用 t 检验比较两个系统的性能。因为需要计算每个数据集的均值,所以在方差公式中的分母和自由度是 $(11 - 1) + (11 - 1) = 20$ 。这些数据并不能确定系统 1 是优越的。尽管在平均值上两个系统是不同的,但是它们的样本方差太高了,以至于不能得出确定的结论

	系统 1	系统 2
得分	71, 61, 55, 60, 68, 49, 42, 72, 76, 55, 64	42, 55, 75, 45, 54, 51 55, 36, 58, 55, 67
总分	609	526
n	11	11
均值 $\bar{x}_i$	55.4	47.8
$s_i^2 = \sum (x_{ij} - \bar{x}_i)^2$	1375.4	1228.8
自由度	10	10

$$\text{合并方差 } s^2 = \frac{1375.4 + 1228.8}{10 + 10} \approx 130.2$$

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s^2}{n}}} = \frac{55.4 - 47.8}{\sqrt{\frac{2 \cdot 130.2}{11}}} \approx 1.56$$

在测试数据上使用留存数据估计

只要 n -gram 的频率 $C(w_1 \cdots w_n)$ 是我们用来预测它在文本中未来频率的惟一条件,就可以根据最大化测试数据概率的原则,在测试数据上使用留存估计法给出概率折扣估计的正确答案。经验估计量测了在训练数据中出现了 r 次的 n -gram 实际上在测试数据中出现了多少次。表 6.4 中的经验估计 $f_{\text{empirical}}$ 是这样得到的,随机地把 AP 语料库中的 44 000 000 个 bigram 划分到相同大小的训练和测试集中,在训练语料中统计 22 000 000 个 bigram 的频率,然后在测试语料中作为留存估计。而表中其他的估计仅仅是从训练语料中得来的,所以经验估计 $f_{\text{empirical}}$ 可以被看成是一种通过参考测试数据得到的经验而确定的黄金标准。

- ① 关于比较统计和机器学习算法的测试方法论的系统讨论可以在 (Dietterich 1998) 中找到。一个好的研究实例可以在 (Mooney 1996) 中找到,这是一个语义消歧的例子。
- ② 根据本书配套网站上提供的勘误表,表 6.6 在计算总分时出现了错误,系统 1 的总分应为 673,系统 2 的总分应为 593。感兴趣的读者可以自己重新计算其他数据。——译者注

6.2.4 交叉验证

上面刚刚讨论的 $f_{\text{empirical}}$ 估计是通过考虑在测试数据中实际发生的事件构成的。留存估计的思想是,把训练数据分成两部分,得到与经验估计相同的效果。使用一部分数据建立最初的模型,然后使用另一部分留存数据来精炼这个模型。这种方法的缺点是,最初的训练数据比较少,所以得到的概率估计也会不太真实。

与这种训练数据的一部分只用来训练,另一部分只用来进行平滑的做法相比,更有效率的方案是,训练数据的每一部分既作为最初的训练数据也作为留存数据。通常,这样方法在统计学上被称为交叉验证(cross-validation)。

Jelinek and Mercer(1985)使用了一种双向交叉验证的方法,称之为删除估计(deleted estimation)。我们设 N_r^a 为 a 部分训练数据出现 r 次的 n -gram 的数量, T_r^a 为这些 a 部分中出现的 bigram 在训练数据的 b 部分出现的总次数。根据基础训练数据的不同,标准的留存估计将会是下面的两者之一:

$$P_{\text{ho}}(w_1 \cdots w_n) = \frac{T_r^{01}}{N_r^0 N} \text{ 或 } \frac{T_r^{10}}{N_r^1 N} \quad \text{其中 } C(w_1 \cdots w_n) = r$$

更加有效的删除插值估计(deleted interpolation estimate)在这两部分分别进行训练和平滑,然后根据 N_r^0 相对于 N_r^1 的比率进行加权平均:

$$P_{\text{del}}(w_1 \cdots w_n) = \frac{T_r^{01} + T_r^{10}}{N(N_r^0 + N_r^1)} \quad \text{其中 } C(w_1 \cdots w_n) = r \quad (6.11)$$

在大的训练语料库中,在训练数据上进行删除估计的效果要好于留存估计,并且在表 6.4 中也显示了删除估计法产生的结果非常接近经验的黄金标准^①。然而,它对低频事件仍然不太适用。这种方法高估了未知事件的期望频率,但是低估了在训练数据中只出现一次的事件的期望概率。把文本划分为两部分,并且假设在 $N/2$ 大小的样本中观察到事件 r 次的概率是在 N 大小的样本中观察概率的两倍,就可以通过观察在 $N/2$ 大小的样本中事件出现的次数来估计事件的概率。但是,随着训练语料库尺寸的增加,在留存数据中遇到未知 n -gram 的比例和对未知 n -gram 的概率估计一般都会减小(但是不可以忽略)。正是由于这个原因,在一个较小的训练语料库上收集到的计数可能导致对未知事件概率的估计偏高。

还有其他一些交叉验证的方法。在实践中,Ney et al.(1997)提出了一种称为 Leaving-One-Out 的方法,这种方法主要的训练语料的大小是 $N - 1$ 个词次(token),而其中一个词次作为一种模拟测试的留存数据。这个过程要重复 N 次,所以数据的每一部分都依次被遗漏。这样的训练体制的优点是,如果数据的任何一个特殊部分没有被观测到,它可以发现模型产生变化的效果。并且 Ney 等人揭示了在这种方法的结论公式和广泛使用的 Good-Turing 方法之间存在着很强的联系^②,下面将介绍 Good-Turing 方法。

① 回忆一下,虽然经验黄金标准是通过留存估计得来的,但是这个留存估计是基于测试语料的! Chen and Goodman (1998)研究发现,对于一个较小的训练语料,留存估计要好于删除估计。

② 然而,Chen and Goodman(1996:314)提出,每次遗漏一个词的做法是有问题的,并且认为在删除插值法中使用大的删除块是更好的选择。

6.2.5 Good-Turing 估计

Good-Turing 估计法

Good(1953)根据图灵机原理提出了一种确定事件频率或者概率估计的方法,假设事件的分布是二项式分布。这种方法适用于从大词表中得来的大量数据观测,而且,尽管词汇和 n -gram 不服从二项式分布,它也可以很好地处理 n -gram 问题。Good-Turing 估计法中的概率估计是: $P_{GT} = r^*/N$, 这里的 r^* 可以认为是一个调整过的频率。对于一个事先观测到的事件,根据 Good-Turing 方法可以得到:

$$r^* = (r + 1) \frac{E(N_{r+1})}{E(N_r)} \quad (6.12)$$

公式中, E 表示一个随机变量的期望。参见(Church and Gale 1991a; Gale and Sampson 1995)关于这个公式出处的讨论。转移到未知事件的总概率是 $E(N_1)/N$ (参见习题 6.5)。

使用经验估计,我们希望用观测到的 N_r 代替 $E(N_r)$ 。但是不能全部这样做,因为对于较大的 r , 经验估计将会变得非常不可信。特别是最经常出现的 n -gram 的估计概率为零,因为出现比这个频率多 1 的 n -gram 的数量为零! 在实际中,有两种解决方法可以使用。一种方法是,使用 Good-Turing 方法只是重新估计 $r < k$ 的频率, k 是一个常量(例如 10)。低频词的数量巨大,所以期望频率的观测频率估计是非常准确的,而且高频词的 MLE 估计也非常准确,可以不用对它们进行折扣。另一种方法是,通过观测值 (r, N_r) 拟合一个函数 S , 使用平滑值 $S(r)$ 代替期望值(这样就导致根据不同的曲线拟合方法可以产生一系列概率值。Good(1953)讨论了几种曲线拟合的方法。转移到未知事件上的概率 N_1/N 可以在未知事件中统一分配,也可以通过更复杂的方法来分配(参见 6.3 节)。对未知事件使用统一分配的方法,可以得到:

Good-Turing 估计法: 如果 $C(w_1 \cdots w_n) = r > 0$

$$P_{GT}(w_1 \cdots w_n) = \frac{r^*}{N} \quad \text{其中 } r^* = \frac{(r+1)S(r+1)}{S(r)} \quad (6.13)$$

如果 $C(w_1 \cdots w_n) = 0$

$$P_{GT}(w_1 \cdots w_n) = \frac{1 - \sum_{r=1}^{\infty} N_r \frac{r^*}{N}}{N_0} \approx \frac{N_1}{N_0 N} \quad (6.14)$$

Gale and Sampson(1995)提出了一种简单有效的方法,对 Good-Turing 方法进行了简化,它有效地混合了上面论述到的两种方法。使用简单的势曲线 $N_r = ar^b$ 作为平滑曲线(在 $b < -1$ 时给出了适合的双曲关系),而且可以通过对数形式的公式 $\log N_r = \log a + b \log r$ 做线性回归(线性回归在 15.4.1 节中涉及,也可以在统计学的书籍中找到)来估计参数 a 和 b 。但是他们指出,这样的简单曲线可能仅仅适合于 r 较高的情况。当 r 值较低时,直接使用 N_r 。在开始时使用直接估计 N_r 来代替期望值,随着频率增加到直接计算的 r^* 值和通过平滑方程计算的值

有很大差别时,再对高频使用平滑估计^①。简化 Good-Turing 方法可以给出非常好的估计值,通过比较表 6.4 中的 Good-Turing 值 f_{GT} 和经验黄金标准就能看到这一点。

在上述这些方法中,都重新归一化(renormalize)了所有的概率估计,以确保得到合理的概率分布结果。为此,我们可以调整转移到未知事件上的概率大小(像式(6.14)一样),或者,更好的方法似乎是保持转移到未知事件上的概率 N_1/N 不变,并且重新归一化所有的已知事件的概率估计(由 Gale and Sampson(1995)提出)。

Austen 语料中的复频

为了做 Good-Turing 估计,第一步就是计算不同频率的频率,又称复计数(count-counts)。表 6.7 显示了 bigram 和 trigram 的一些复频分布结果。这些数字是 1.4.3 节中 Zipf 分布中的数据,但是结构细节不同,而且更被夸大了,因为是词序列的计数。表 6.8 给出了重估的频率 r^* 和 bigram 的概率。

表 6.7 节选自 Austen 语料库中 bigram 和 trigram 的复频(frequencies of frequencies)分布

bigram				trigram			
r	N_r	r	N_r	r	N_r	r	N_r
1	138 741	28	90	1	404 211	28	35
2	25 413	29	120	2	32 514	29	32
3	10 531	30	86	3	10 056	30	25
4	5997	31	98	4	4780	31	18
5	3565	32	99	5	2491	32	19
6	2486		...	6	1571		...
7	1754	1264	1	7	1088	189	1
8	1342	1366	1	8	749	202	1
9	1106	1917	1	9	582	214	1
10	896	2233	1	10	432	366	1
	...	2507	1		...	378	1

对于 bigram 来说,转移到未知事件上的概率是 $N_1/N = 138\,741/617\,091 = 0.2248$ 。bigram 的空间是词表大小的平方,我们共观测到 199 252 个 bigram,所以如果在未知事件上使用一致估计,每一个未知 bigram 的概率估计是: $0.2248/(14\,585^2 - 199\,252) = 1.058 \times 10^{-9}$ 。如果我们现在希望得到 bigram 模型的条件概率估计,对于 bigram 的概率估计使用 Good-Turing 估计,对于 unigram 的估计使用 MLE 估计,那么可以得到下式:

$$P(\text{she}|\text{person}) = \frac{f_{GT}(\text{person she})}{C(\text{person})} = \frac{1.228}{223} = 0.0055$$

使用这种方法得到的结果显示在表 6.9 中,可以和表 6.3 中的 bigram 估计进行比较。这些概率估计一般来看是非常合理的。把这些数字相乘,得到了一个 1.278×10^{-17} 的概率估计。这个概率至少远大于 ELE 的估计值,但是它仍然使用了未知 bigram 上概率统一分布的假设。

① 如果差异超过 Good-Turing 估计的标准差的 1.65 倍,我们认为 r^* 的估计有很大的差异,Good-Turing 估计的标准差由下式给出:

$$\sqrt{(r+1)^2 \frac{N_{r+1}}{N_r^2} \left(1 + \frac{N_{r+1}}{N_r}\right)}$$

表 6.8 Good-Turing 估计是经过平滑的,使用配套网站上提供的程序

r	r^*	$P_{GT}(\cdot)$
0	0.0007	1.058×10^{-9}
1	0.3663	5.982×10^{-7}
2	1.228	2.004×10^{-6}
3	2.122	3.465×10^{-6}
4	3.058	4.993×10^{-6}
5	4.015	6.555×10^{-6}
6	4.984	8.138×10^{-6}
7	5.96	9.733×10^{-6}
8	6.942	1.134×10^{-5}
9	7.928	1.294×10^{-5}
10	8.916	1.456×10^{-5}
...		
28	26.84	4.383×10^{-5}
29	27.84	4.546×10^{-5}
30	28.84	4.709×10^{-5}
31	29.84	4.872×10^{-5}
32	30.84	5.035×10^{-5}
...		
1264	1263	0.002 062
1366	1365	0.002 228
1917	1916	0.003 128
2233	2232	0.003 644
2507	2506	0.004 092

表 6.9 Persuasion 语料句子中的 Good-Turing bigram 频率估计

$P(\text{she} \text{person})$	0.0055
$P(\text{was} \text{she})$	0.1217
$P(\text{inferior} \text{was})$	6.9×10^{-8}
$P(\text{to} \text{inferior})$	0.1806
$P(\text{both} \text{to})$	0.000 395 6
$P(\text{sisters} \text{both})$	0.003 874

6.2.6 简要注释

Ney and Essen(1993)和 Ney et al.(1994)提出了两种折扣方法。在绝对折扣模型中,所有的非零 MLE 频率用一个小的常数 δ 折扣,由此得到的频率增益被平均分配到未知事件上。

绝对折扣(Absolute discounting):如果 $C(w_1 \cdots w_n) = r$,

$$P_{\text{abs}}(w_1 \cdots w_n) = \begin{cases} (r - \delta)/N & \text{若 } r > 0 \\ \frac{(B - N_0)\delta}{N_0 N} & \text{其他} \end{cases} \quad (6.15)$$

回忆前述内容, B 是类别数目。在线性折扣方法中,非零的 MLE 频率乘以一个稍微小于 1 的常量,留出的概率仍分配到未知事件上。

线性折扣 (Linear discounting): 如果 $C(w_1 \cdots w_n) = r$

$$P(w_1 \cdots w_n) = \begin{cases} (1 - \alpha)r/N & \text{若 } r > 0 \\ \alpha/N_0 & \text{其他} \end{cases} \quad (6.16)$$

这些估计方法和频率移动的作用是等同的,用一些小数字 ϵ 代替未知事件的零概率,并且重新调整其他的概率,使得概率总和为 1(在调整方法中选择减一个常量或者乘以一个常量,对概率进行折扣)。重新考虑表 6.4 中的数据,可以看到绝对折扣似乎可以提供一个好的估计。比较表中 $f_{\text{empirical}}$ 的数据,似乎除了只出现一次的 bigram(这些 bigram 会被估计偏低)外, $\delta \approx 0.77$ 的折扣效果都很好。通常我们使用留存数据来估计合适的 δ 值。绝对折扣法的扩展也很成功,我们将会在下面讨论。很难证明线性折扣法的正确性。因为一般来看,训练文本中事件频率越高,没有调整过的 MLE 估计的准确性就越高,但是线性折扣的方法恰好不会近似这个观测值。

Lidstone 法则的缺点是它依赖于模型的类别数目。虽然一些空的类别是由数据稀疏问题引起的,但是大部分空类别是由原理上的缺陷引起的。Good-Turing 估计对已知事件的概率估计却不依赖分类的数目。Ristad(1995)提出一个假设,自然序列只使用可能类别的子集。对于演替的自然法则(Natural Law of Succession),得出了很多形式,包括对观测频率 $C(w_1 \cdots w_n) = r$ 的 n -gram 的概率估计形式:

$$P_{\text{NLS}}(w_1 \cdots w_n) = \begin{cases} \frac{r+1}{N+B} & \text{若 } N_0 = 0 \\ \frac{(r+1)(N+1+N_0-B)}{N^2+N+2(B-N_0)} & \text{若 } N_0 > 0 \text{ 且 } r > 0 \\ \frac{(B-N_0)(B-N_0+1)}{N_0(N^2+N+2(B-N_0))} & \text{其他} \end{cases} \quad (6.17)$$

这个法则的主要特征是:(i) 如果有些事件在每个分类中都能看到,那么它减弱了 Laplace 法则;(ii) 被指定到未知事件上的概率大小以 N 的平方减小;(iii) 被指定到未知事件上的总的概率 B 不依赖于分类的数目,所以这种方法也适用于大的词表。

6.3 组合估计法

迄今为止,我们考虑过的方法都只利用了 n -gram 的频率 r 一项数据,试图从这个频率产生 n -gram 在更多文本中出现概率的最佳估计。但是对于所有没有出现过的或者很少出现的 n -gram,给定相同的概率估计不是一个好的解决方法,我们希望通过考虑 n -gram 中 $(n-1)$ -gram 的频率产生一个更好的概率估计。如果 $(n-1)$ -gram 本身就很少出现,就给 n -gram 一个低的估计值。如果 $(n-1)$ -gram 有一个中等频率,就给 n -gram 一个较高的概率估计^①。Church and Gale (1991a)给出了这种思想的细致研究,显示了未知的 bigram 的概率估计和组成它的 unigram 的概率之间的联系。对于未知的 bigram,计算了独立连接(joint-if-independent)概率 $P(w_1)P(w_2)$,然后根据这个概率对 bigram 进行分类。每个类中使用 Good-Turing 估计法给出修正频率,然后用这些修正过的频率产生概率。

但是在这一节中我们考虑的是更一般化的问题:怎样从不同的模型中组合多重概率估计。

① 但是,如果 $(n-1)$ -gram 出现得非常频繁,那么我们实际上给 n -gram 的概率也非常小,因为 n -gram 没有出现而 $(n-1)$ -gram 出现得非常频繁的情况表明 n -gram 概率小是应该的。

如果有几个利用历史预测的模型,那么我们可能希望把它们组合起来产生一个更好的模型。可以通过平滑或组合不同的信息资源的方法来实现这个想法。

对 n -gram 模型来说,找到合适的组合不同阶模型的方法是成功的关键。一种合并不同 n 值的 MLE n -gram 估计(对未知词有一定的概率转移)的方法使用了简单的线性插值技术(下面将会介绍),得出了一个非常好的语言模型(Chen and Goodman 1996)。实际上我们可以做得更好,而不是简单地使用上述方法。我们需要把上述方法和下面将要提供的方法组合起来。

6.3.1 简单的线性插值

解决 trigram 模型中数据稀疏问题的一种方法是,把 bigram 模型和 unigram 模型组合到 trigram 模型中,这两个模型可以在一定程度上容忍数据稀疏问题。这样的情况下肯定会有多个概率估计出现,我们可以把它们线性组合起来,对每个概率分布加权就可以得到新的概率方程。在统计自然语言处理中,这种方法通常被称为线性插值法,在其他的地方则常常被称为(最终)混合模型(mixture model)。当所有被插值的方程都使用最佳分辨函数的条件信息子集时(如 trigram, bigram 和 unigram 模型的组合形式),这种方法经常被认为是删除插值法(deleted interpolation)。对于插值的 n -gram 语言模型,例如一个 trigram 模型中的删除插值法,最基本的做法是:

$$P_{\text{li}}(w_n | w_{n-2}, w_{n-1}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n | w_{n-1}) + \lambda_3 P_3(w_n | w_{n-1}, w_{n-2}) \quad (6.18)$$

其中, $0 \leq \lambda_i \leq 1$ 并且 $\sum_i \lambda_i = 1$ 。虽然说这些权值可以手工设定,但是通常希望得到权值的最佳组合。通过 EM(Expectation Maximization)算法的一个简单应用,可以自动做到这一点,详细情况将在 9.2.1 节中讨论。也可以使用其他很多算法来实现上述想法。例如,Chen and Goodman(1996)使用了在(Press et al. 1988)中给出的 Powell 算法。Chen and Goodman(1996)证明这个简单模型(只是对未知历史的处理稍微复杂一些,并且对未登录词预留了一定的概率)的效果非常好,这个模型被用做实验中的底线模型(baseline model),参见 7.1.3 节。

6.3.2 Katz 回退算法

在回退模型中,要根据不同模型的特性来使用它们。最详细的模型可以提供足够的关于上下文的可信信息。另外,回退还可以用来进行平滑或者组合信息源。

n -gram 的回退模型是 Katz(1987)提出的。一个 n -gram 的概率估计允许回退到越来越短的历史上:

$$P_{\text{bo}}(w_i | w_{i-n+1} \cdots w_{i-1}) = \begin{cases} (1 - d_{w_{i-n+1} \cdots w_{i-1}}) \frac{C(w_{i-n+1} \cdots w_i)}{C(w_{i-n+1} \cdots w_{i-1})} & \text{若 } C(w_{i-n+1} \cdots w_i) > k \\ \alpha_{w_{i-n+1} \cdots w_{i-1}} P_{\text{bo}}(w_i | w_{i-n+2} \cdots w_{i-1}) & \text{其他} \end{cases} \quad (6.19)$$

如果关注的 n -gram 出现的次数大于 k 次(k 在正常情况下设为 0 或 1),那么就使用 n -gram 的估计方法,就像公式中的第一行一样。但是 MLE 估计要被一个确定的数量折扣(由方程中的 d 来表示),所以一些概率将会被转移到未知的 n -gram 上,这些未知的 n -gram 的概率也要通过回退来估计。MLE 估计需要以某种形式进行折扣,否则将不会有概率被分配到低阶的模型上。计算折扣的一个方法就是上面讨论的 Good-Turing 估计法,并且这也是 Katz 实际使用的方

法。如果这个 n -gram 在训练语料中没有出现或者出现次数小于等于 k 次,那么我们将使用一个短一些的 n -gram 概率估计。但是这个回退概率要乘以一个正规化因子 α ,以确保在折扣过程中预留的概率会分配到以回退方法估计的 n -gram 中。要注意一种特殊情况,历史中的 $(n-1)$ -gram 也是未知的,那么对于任何一个 w_i ,公式的第一行都是不可用的,同时回退因子 α 的值要设为 1。如果选择了公式的第二行,我们要通过估计 $(n-1)$ -gram 的概率来递归地得到 n -gram 的估计概率。这个递归过程可以继续下去,所以我们可以从一个 four-gram 模型开始,结束于一个 unigram 模型,使用 unigram 的频率估计下一个词的概率。

虽然通常回退方法在缺少很多数据的情况下是合理的,但事实上在有些环境中它的效果非常差。例如,如果我们已经看到 $w_i w_j$ 出现很多次,并且 w_k 是一个常用词,但是我们从来没有看到 trigram: $w_i w_j w_k$,那么在这种情况下实际上已经可以得出一个重要的结论了,这个 trigram 很可能是一个“零概率”的例子,而不是例行公事地使用回退算法,通过 bigram 估计值 $P(w_k|w_j)$ 来估计 $P(w_k|h)$ 。Rosenfeld and Huang(1992)提出了一个更加复杂的回退模型,试图来解决这个问题。

回退模型有时会受到批评,因为在把一些数据添加到训练语料中时它的概率估计会突然改变,新添加的数据可能导致回退模型选择不同阶的 n -gram 模型作为估计概率的基础模型。不过总的来说它还是一种简单方法,并且在实际中效果良好。

6.3.3 一般化线性插值

在简单的线性插值中,权值仅仅是一个单个数字,可以定义一个更加一般化并有效的模型,它的权值是一个关于历史的函数。对于 k 个概率方程 P_k ,线性插值模型的一般化形式是:

$$P_H(w|h) = \sum_{i=1}^k \lambda_i(h) P_i(w|h) \quad (6.20)$$

在公式中,任取 h , 都有 $0 \leq \lambda_i(h) \leq 1$ 且 $\sum_i \lambda_i(h) = 1$ 。

线性插值法被经常使用是因为它是一种非常普通的组合模型的方法。随机加入一个未知模型到线性插值中,倘若使用 EM 算法对这个模型可以找到一个好的权值,就不会对系统产生危害。但是线性插值法也可以使组合模型的效果变差,特别是带有不同权值的历史划分不精确的情况下。例如,如果 λ_i 在 n -gram 模型的线性插值中是不变的,那么 unigram 估计就总是带有相同权值被组合到插值模型中,无论 trigram 估计是非常好(由于有大量的数据)还是非常坏。

一般来说,不会根据单独的历史来设置权值。对每一个 $w_{(i-n+1)(i-1)}$ 都训练一个不同的 $\lambda_{w_{(i-n+1)(i-1)}}$ 是极为不当的,因为这样做会恶化数据稀疏的问题。所以有些人希望使用一些历史等价类来设置 λ 。Bahl et al.(1983)建议根据 $C(w_{(i-n+1)(i-1)})$ 把 λ 划分到的等价类,把所有具有相同频率历史的参数都绑定在一起。

Chen and Goodman(1996)提出把 λ 参数划分到等价类并不是一个好的方法,更好的方法是根据非零元素频率的平均数来分组 λ :

$$\frac{C(w_{(i-n+1)(i-1)})}{|w_i : C(w_{(i-n+1)i}) > 0|} \quad (6.21)$$

也就是说,在 n -gram: $w_{i-n+1} \cdots w_{i-1} w_i$ 的非零频率上得到平均值。这样做的理由是因为语言的句法结构,哪个词可能或者通常会出现其他词的后面是有很强的结构限制的。虽然大部

分统计自然语言处理语言模型的核心假设是任何词都可以出现在任何其他词的后面(这样做会使我们不得不处理所有可能的选择),不过在很多情况下由于语法的限制,我们预测的下一个词会有很强的限制。虽然有些 n -gram 仅仅是没有看到,但是其他一些 n -gram 对于创建一个词组来说是“空语法”(grammatical zeroes),因为这些 n -gram 不符合语言规则。例如,在我们的 Austen 训练语料库中,两个 bigram: *great deal* 和 *of that* 都出现了 178 次。但是,在语料中 *of that* 后面跟随了 115 个不同的词,平均值是 1.55,这反映了这样的事实,任何副词、形容词和名词都可以恰当地出现在名词词组中,而且任何一个以大写字母开头的词作为一个句子的开始都是可能的。因此这种情况下的空语法现象就特别少(主要是动词和介词)。在另一方面, *great deal* 后面仅仅出现了 36 个词,得到的平均值是 4.94。虽然新句子的开始仍然是随机的,但是可能的选择已经被极大地限制到了连词、介词和形容词比较级上。特别是,介词 *of* 的出现占据了总次数的 38%。这个比较高的平均值表明在这个 bigram 后面跟随了大量的空语法,所以在这样的上下文中给未出现的 n -gram 一个非常低的估计是合理的。

最后要注意到,回退模型实际上就是一般线性插值模型的特殊情况。在回退模型中,已经被回退模型选择的模型系数 $\lambda_i(h)$ 是 1,其他系数是 0。

6.3.4 简要注释

Bell et al. (1990) 和 Witten and Bell (1991) 介绍了一系列平滑算法,用来压缩文本的规模。他们提到的“方法 C”通常被称为 Witten-Bell 平滑法,已经在语音语言模型的平滑中广泛应用。这个方法的思想是,对于一个先前没有看到的事件,使用在训练语料中看到这样一个新事件的概率来代替它的概率。实际上这个概率是根据某一个历史得出的。比如,为了计算看到一个新词出现在 *sat in* 后面的概率,我们计算在训练语料中一个新词出现在 *sat in* 后面的频率,这也就是在训练数据中以 *sat in* 开头的 trigram 的数目。因此这种方法也是一个普通线性插值的例子:

$$P_{WB}(w_i | w_{(i-n+1)(i-1)}) = \lambda_{w_{(i-n+1)(i-1)}} P_{MLE}(w_i | w_{(i-n+1)(i-1)}) + (1 - \lambda_{w_{(i-n+1)(i-1)}}) P_{WB}(w_i | w_{(i-n+2)(i-1)}) \quad (6.22)$$

公式中分配到新的 n -gram 上的概率是通过下式给出的:

$$(1 - \lambda_{w_{(i-n+1)(i-1)}}) = \frac{|\{w_i : C(w_{i-n+1} \cdots w_i) > 0\}|}{|\{w_i : C(w_{i-n+1} \cdots w_i) > 0\}| + \sum_{w_i} C(w_{i-n+1} \cdots w_i)} \quad (6.23)$$

然而,Chen and Goodman (1998) 的结果指出,这种方法作为语言模型的平滑技术并没有我们本章中讨论的其他方法好(特别是使用小训练集时效果很差)。

Samuelsson (1996) 提出了线性连续抽象法 (Linear Successive Abstraction), 用来确定删除插值类模型的参数,不需要在留存数据上进行经验确定。Samuelsson 的结果指出,在词性标注任务中它的效果和传统删除插值方法的效果类似;但是我们不太清楚这种方法在词的 n -gram 模型中的评价。

Chen and Goodman (1996) 考察了另一种简单而有效的平滑方法。MacKay and Peto (1990) 提出了平滑分布形式:

$$P_{MP}(w_i | w_{i-n+1} \cdots w_{i-1}) = \frac{C(w_{i-n+1} \cdots w_i) + \alpha P_{MP}(w_i | w_{i-n+2} \cdots w_{i-1})}{C(w_{i-n+1} \cdots w_{i-1}) + \alpha} \quad (6.24)$$

在这里 α 表示 Lidstone 法则中的加数,但是它的分布是低阶分布。Chen and Goodman(1996)指出 α 和只出现一次的词的数量之间存在比例关系,并且得出:

$$\alpha = \gamma(N_1(w_{i-n+1} \cdots w_{i-1}) + \beta) \quad (6.25)$$

在这里, $N_1(w_{i-n+1} \cdots w_{i-1}) = |\{w_i: C(w_{i-n+1} \cdots w_{i-1}) = 1\}|$, 并且 β 和 γ 可以在留存数据上进行最优化。

Kneser and Ney(1995)提出了一种基于扩展绝对折扣方法的回退模型,它给出了一种更准确的方法来确定回退的分布。Chen and Goodman(1998)发现这种方法和它的扩展方法都具有非常出色的平滑效果。

6.3.5 Austen 语料库的语言模型

随着插值和回退方法的引入,我们已经可以对 Austen 语料库构建一个非常好的语言模型了。我们使用 CMU-剑桥统计语言建模工具(可通过本书配套网站获得)建立的回退语言模型基本上依据 Katz(1987)^① 的方法。然后,我们在测试集 Persuasion 上计算了这些语言模型的交叉熵(和混乱度)。结果示于表 6.10 中。先前给出的例句中下一个词的估计概率和用来估计的 n -gram 大小在表 6.11 中列出。最终得到的句子概率估计至少高于开始讨论的 unigram 估计。

表 6.10 使用 Good-Turing 估计的回退语言模型在 Persuasion 上的测试结果

模型	交叉熵	混乱度
Bigram	7.98 比特	252.3
Trigram	7.90 比特	239.1
Four-gram	7.95 比特	247.0

表 6.11 对应于不同语言模型的测试子句的估计概率。其中 unigram 估计是我们先前的 MLE unigram 估计。其他的两个估计使用了回退语言模型。表中最后一列给出了子句总的概率估计

	$P(\text{she} h)$	$P(\text{was} h)$	$P(\text{inferior} h)$	$P(\text{to} h)$	$P(\text{both} h)$	$P(\text{sisters} h)$	乘积
Unigram	0.011	0.015	0.000 05	0.032	0.0005	0.0003	3.96×10^{-17}
Bigram	0.005 29	0.1219	0.000 015 9	0.183	0.000 449	0.003 72	3.14×10^{-15}
n 的选用	2	2	1	2	2	2	
Trigram	0.005 29	0.0741	0.000 016 2	0.183	0.000 384	0.003 23	1.44×10^{-15}
n 的选用	2	3	1	2	2	2	

虽然总的来说,在测试数据上 trigram 模型的效果要好于 bigram 模型,但是我们注意到在例句中 bigram 模型实际上有更高的概率。而且, four-gram 模型的效果要稍微比 trigram 模型差,这是由于训练数据规模较小引起的。回退模型在简单忽略不合适的长上下文时一般不是最好的方法,并且如果相对于训练语料的规模,用来构建模型的 n -gram 太长,这个模型的效果就会趋向于恶化。

6.4 结论

很多平滑方法都是可用的,它们通常会给出相似的并且比较好的效果。使用 Good-Turing

^① Good-Turing 平滑的一个版本,仅仅折扣低频,即出现次数小于 7 次的词。

估计和线性插值法或者回退算法可以避开数据稀疏的问题。Chen and Goodman (1996, 1998) 给出了不同平滑算法的广泛评价。(Chen and Goodman 1998) 的结论认为他们开发的 Kneser-Ney 回退平滑的变形形式通常会得到最好的效果。Church and Gale (1991a) 声称使用 Good-Turing 平滑方法在一个 2 000 000 个词以上的文本中训练得到的 bigram 模型的效果比它更好, 而且还可以假设在两倍数量级文本上训练的 trigram 模型也符合上述结论。但是在所有其他情况下, Chen 和 Goodman 的方法似乎是最好的。虽然简单的平滑方法对于探索性研究是合适的, 但是如果你希望得到一个带有最优化结果的系统, 最好避免使用这些简单的平滑方法。现在的研究都是在组合概率模型和处理稀疏数据方面进行的。

6.5 深入阅读

在语言模型上下文的统计估计方面, 重要的研究工作包括 (Katz 1987), (Jelinek 1990), (Church and Gale 1991a), (Ney and Essen 1993) 和 (Ristad 1995)。关于概率估计技术的其他讨论可以在 (Jelinek 1997) 和 (Ney et al. 1997) 中找到。Gale and Church (1994) 给出了“加 1”问题的细节论述。Good-Turing 估计的简单解释可以在 (Gale and Sampson 1995) 中找到。在 (Chen and Goodman 1996, 1998) 中比较了各种平滑算法, 我们在这里为读者特别推荐这两篇论文。

使用最大似然估计参数值的想法最先在 (Fisher 1922) 中提出。在 (Ney et al. 1997) 中可以找到相对频率实际上就是最大似然估计的证明。

当前, 最大熵方法被广泛应用于模型组合。我们把最大熵模型的讨论放到了第 16 章中。语言模型的应用可以参见 (Lau et al. 1993) 和 (Rosenfeld 1994, 1996)。

6.2.2 节中引用的早期工作出现在 (Lidstone 1920), (Johnson 1932) 和 (Jeffreys 1948) 中。(Ristad 1995) 讨论了折扣问题。Good (1979: 395 ~ 396) 讨论了 Good-Turing 平滑方法的最初想法。这篇论文扩充后再版于 (Britton 1992)。

6.6 习题

6.1 [★★] 研究测试数据 (不同于训练数据) 中未知 n -gram 的百分比。研究当下面的条件部分或者全部改变时的这个百分比的变化: (i) 模型的阶次 (也就是 n); (ii) 训练数据的大小; (iii) 训练数据的类别; (iv) 在类别、领域和年代上测试数据和训练数据的相似度。

6.2 [★] 作为 Laplace 法则问题的小规模的例子, 试计算使用 Laplace 法则给出的概率估计: 给出含有 1000 项词条的 100 个样本, 在这些样本中, 9 项词条出现了 10 次, 2 项词条出现了 5 次, 其他 989 项词条没有出现。

6.3 [★] 证明使用 ELE 会使下面的概率方程成立:

$$\sum_{w_1 \cdots w_n} P_{\text{ELE}}(w_1 \cdots w_n) = 1$$

6.4 [★] 使用下面给出的 Austen 测试语料中的词和 bigram 频率, 验证 6.2.2 节给出的例句 *she was inferior to both sisters* 的 ELE 估计 (使用在语料中 *she* 的前面是 *person* 的事实)。

w	$C(w)$	$w_1 w_2$	$C(w_1 w_2)$
person	223	person she	2
she	6 917	she was	843
was	9 409	was inferior	0
inferior	33	inferior to	7
to	20 042	to both	9
both	317	both sisters	2

6.5 [★]证明Good-Turing估计法是有根据的,也就是证明:

$$\sum_{w_1 \cdots w_n} P_{GT}(w_1 \cdots w_n) = \frac{f_{GT}(w_1 \cdots w_n)}{N} = 1$$

6.6 [★]对于例句 *she was inferior to both sisters* 我们已经使用 bigram 模型计算了Good-Turing概率估计,而对未出现的bigram使用了一致估计。确定你可以重复这些过程,并且试着使用trigram模型重复上面的过程。它的效果怎么样?

6.7 [★★]使用本书配套网站上的软件(或者可以自己实现)为一个语料库构建一个语言模型。当使用交叉熵来评价模型的效果时,实验哪些选项可以给出最好的语言模型。

6.8 [★★]分别从两个不同的领域中抽取两个语料库,并且把它们中的每一个都划分成训练集和测试集。对每个领域在训练数据上构建语言模型。然后,对于测试集使用两个不同的语言模型计算交叉熵的数值。看看交叉熵估计有什么不同。

6.9 [★★]写一个程序,它可以学习某些文本的 n -gram 词模型(可以做平滑,但是在本习题中并不是必要的)。在几个世界新闻网的文章上或者其他不同类别的文本上分别训练模型,然后基于这些模型产生一些随机的文本。不同的 n 值输出的文本的可理解性怎么样? 在产生的文本中,是否清楚地保留了各种新闻的不同特性?

6.10 [★★]写一个程序,基于在已知语言文本上的训练,可以识别一个短的文本片段中的语言是什么语言。例如,下面每一行都是不同语言的文本:

doen is ondubbelzinnig uit
 prétendre à un emploi
 uscirono fuori solo alcune
 look into any little problem

如果读者稍微有一点欧洲语言的知识,就可能识别出每个例子是什么语言。这是一个分类任务,在这个任务中可以使用一些本章中讨论的语言模型技术。提示:考虑字母 n -gram 和词 n -gram。这是一个别人已经研究过的问题,参见(Dunning 1994)。网页上有很多现有语言识别系统,包括最初对这个习题的解决方法。

第7章 语义消歧

本章将对统计自然语言处理中的语义消歧问题进行一般性的概述,并介绍几种最重要的语义消歧算法,描述它们的资源需求和算法性能。

什么是语义消歧?很多词语都具有几个意思或语义,如果把这样的词从上下文中独立出来考虑,就会产生语义歧义,语义消歧就是为了解决这样的歧义问题。作为歧义问题的第一个例子,我们考虑 Webster's New Collegiate Dictionary (Woelf 1973)中的单词 *bank* 和它的两个语义:

- the rising ground bordering a lake, river, or sea...
- an establishment for the custody, loan exchange, or issue of money, for the extension of credit, and for facilitating the transmission of funds

消除歧义的任务就是确定一个歧义词的哪一种语义在一个特殊的使用环境中被调用。通过考虑词汇使用的上下文可以确定其具体的语义。

语义消歧的相关文献对这个问题都有所涉及,即假设某个词具有有限个独立的语义,其语义定义由词典、类义辞典或者其他的参考资源给出,消歧的任务就是根据具体的上下文确定某一个歧义词的确切含义。但是这种对语义消歧任务的叙述是很不恰当的。词汇 *bank* 似乎是歧义词汇最著名的例子,但实际上它并不具备很强的典型性。更典型一些的情况是,歧义词汇具有多种稍微有些联系的语义,而且不清楚是否使用和在哪里使用这些语义。例如,考察词汇 *title*。我们在词典中找到的一些语义是:

- 书籍、法令、艺术或音乐作品等的名称/开头
- 电影开始时的字幕
- 合法的(土地)所有权
- 证明这个权利的文档
- 人名上附加的尊称
- 一种书面作品(使用提喻法,也就是使用部分来表达整体)

简单的处理方法是把一个词的语义定义为一部特殊词典给出的意思。然而,从科学的角度来看这是不合适的,因为不同的词典列出的语义类和数目通常有很大的差异,这不仅仅是因为综合类的词典会有更全面的语义,而且在不同的词典中词汇的基本用法也会被合并到语义中。这些合并通常是任意的。例如,上面的语义列表中区别了两个语义,一个是财产的合法所有权,一个是显示这种所有权的文档。然而,在概念和显示概念的东西之间的语义扩展模式已经是非常广泛的用法,已经可以和其他的用法区别开。例如,当我们说关于 *title of painting* 的事情时,存在相同的歧义。比如,有人在画廊说:

This work doesn't have a title. (7.1)

这个句子可以被理解为这幅画没有题目,或者是给出题目的小布告没有了,这种布告通常出现在画廊中绘画作品的旁边。还有一点不太清楚的是,为什么书籍、法令、艺术或音乐作品被放

到一个语义中,而电影却单独分离出来。在上面的列表中,第二个语义是第一个语义的特例。在很多词典中这是非常常见的现象,如果一个语义是其他语义的特例并且它在文本中频繁使用,那么它可以被单独列出作为一个语义项。上述困难表明,对于大部分词汇来说,它们的用法和语义定义并不是想像中餐桌上你要选择的 5 种奶酪,而更像是一盘烩菜,它们之间有一些可以清晰分辨的内容,但是大部分内容是彼此不确定的,并且是混合在一起的。

尽管这些都是哲学上的问题,但是消除歧义在自然语言处理的很多应用中是非常重要的内容。英语到德语的自动翻译系统对于上面给出的第一个语义(*ground bordering a lake or river*)需要把 *bank* 翻译成 *Ufer*,对于第二个语义要翻译成 *Bank* (financial institution)。信息检索系统回答一个关于“financial bank”的查询时,应该仅仅返回使用 *bank* 的第二个语义的文档。只要系统的反应依赖于处理文本的意思,那么消除歧义就是有益的和非常必要的。

一个词具有不同的词性是另外一种歧义。例如, *butter* 可以是一个名词,也可以是一个动词,比如在句子 *You should butter your toast* 中的用法。确定一个词的具体词性是标注的任务,我们将在第 10 章中讨论这个问题。这两种歧义的概念有什么不同呢? 一个词作为名词而不是动词是有很大区别的,而且这两种用法也有不同的语义,所以同一词汇的不同词性的识别问题可以被看成是一个语义消歧问题。相反,识别语义也可以看成是一种标注问题,但要使用语义标记而不是使用词性标记。在实践中,这两个概念还是有区别的,部分是因为问题本质的区别,部分是因为处理这些问题的方法不同。通常,邻近的结构信息大多数是用来确定词性的(例如,前面一个词是限定词吗?),但是它们一般不被用来确定语义。相反,一个相隔很远的实词对于确定语义是很有效的,但是很少使用它来确定词性。因此,大部分的词性标注模型简单地使用当前上下文,而语义消歧模型通常试图使用规模广泛一些的上下文中的实词。

歧义和消除歧义的内容在很大程度上依赖于用来训练语义消歧系统的文本内容。在首先介绍方法论的第一节后,本章用三个主要的小节来介绍三种不同的消歧方法。7.2 节介绍了有监督消歧(*supervised disambiguation*),它是一种基于标注训练集的消歧。7.3 节介绍基于词典的消歧(*dictionary-based disambiguation*),这种消歧是建立在词典资源上的,比如词典和类义辞典。7.4 节介绍了无监督消歧(*unsupervised disambiguation*),这种情况中未标注文本将应用到训练里。最后我们对语义概念进行深入的讨论,并指出需要深入阅读的材料。

7.1 预备知识

在语义消歧中经常会提及几类重要的方法。它们通常是和自然语言处理相关的,也是我们在这里重点讨论的内容。这些方法包括:有监督和无监督学习,语义消歧中被称为伪词(*pseudoword*)的人工评估数据(*artificial evaluation data*)的使用,如何计算消歧算法性能的上下界从而更好地衡量算法的性能。

7.1.1 有监督和无监督学习

根据有监督或无监督学习(Duda and Hart 1973:45),我们把很多算法进行分类。它们的区别是,在有监督学习中训练数据是已知的(在这里是语义标注),而在无监督学习中训练事例中数据的分类是未知的。因此,无监督学习通常被称为聚类任务(*clustering task*)(参见第 14 章),

而有监督学习通常被称为分类任务(classification task)(参见第16章),也可以称为函数拟合(function-fitting),就是说基于一些数据点推断出函数的形态。

然而,在统计自然语言处理领域中,事情往往并非这么简单。由于标注好的语料非常昂贵,所以人们希望可以从未标注数据中学习,并且试图在自己的算法中使用各种知识资源,比如词典,或者使用更加结构化的数据,比如对齐的双语数据。在一些方法中,系统最初可以建立在一些所谓的“种子”数据集上,然后即可利用这个系统,从未标注数据中学习,逐步扩大种子数据集。与其非要对不同的方法进行分类,还不如简单回答下面的问题更有意义:使用这种方法需要什么样的知识资源?就像我们将要看到的一样,有时不同的知识资源的混合也可以给出相似的信息(例如,使用对齐双语语料,或者使用单语语料和双语辞典)。

7.1.2 伪词

为了在一个自然的歧义词上测试消歧算法的效果,对所需的大量测试数据不得不进行手工消歧,这是一项费时费力的任务。在测试数据很难得到的情况下,通常我们会方便地产生一些人工数据,用来比较和提高算法的性能。在语义消歧的情况下,这些人工数据被称为伪词。

Gale et al. (1992a) 和 Schütze (1992a) 说明了怎样创建伪词(也就是人工歧义词),采用的方法是合并两个或多个自然词汇。例如,创建伪词 *banana-door*,可以在语料库中用伪词 *banana-door* 代替所有出现的 *banana* 和 *door*。这样的伪词既回避了手工标注的困难,又可以为消歧问题轻松创建大规模的训练和测试集:带有伪词的文本作为歧义源文本,最初的文本作为消歧后的文本。

7.1.3 算法性能的上界和下界

衡量某人的算法的效能是非常重要的,但是不讨论这个算法相对于任务的困难程度的效能评价而仅做数字上的评估,是没有任何意义的,即不同的自然语言处理任务其实现难度是有差别的。例如,对于英语文本的词性标注任务,90%的准确率是很容易达到的,远远超出了任何现有机器翻译系统的性能。为了理解系统效能的意义(Gale et al. 1992a),对于一个算法效能的上界和下界进行评价是很有必要的。对于自然语言处理中的许多任务,这是一个很好的想法,特别是在没有标准测试集合来比较不同系统的性能时。

上界(upper bound)通常是指人工的效能。在语义消歧的情形中,如果人工对于一个特殊上下文的语义指代都判断错误,那么就不要指望自动程序会做得更好。如果消歧算法使用受限上下文,例如仅仅考虑歧义词两边的三个词语,确定算法上界就很有益处。在这样的情况下,算法效果不好的原因可能是上下文表示的信息太少,以至于人类基于这样的信息都很难消除歧义。在相同的上下文限制的情况下,通过和人工效能相比较,就可以得到系统效能的评测^①。

Gale et al. (1992a) 中提出了一个语义消歧的上界,测试了下面的任务:给出一个词汇的两个实例,确定它们是否具有一样的语义。这个任务的上界在97%到99%之间。但是Gale的测试集中的大部分词只有很少的语义类别或者有两个有明显区分的语义。而事实上很多歧义词(特别是高频词)都和我们举的例子(*title*)相似,也就是说,它们的语义是相互关联和相互交叉的。相互判断(inter-judge)的准确性依赖于歧义的类型:对于有清晰区分语义的词比较高(大

^① 尽管受到人工上下文的限制,计算机抽取有用预测信息也可能比人工抽取的效果更好。

于 95%), 对于有很多相关语义的多义词比较低(大约在 65% 到 70% 之间)^①。消歧是一个二元判断问题时要比是一个任意的聚类问题时简单。

这就意味着我们不得不考虑每个歧义词的特性, 来确定一个消歧算法是否可以很好地处理这个词。对于像 *bank* 这样的不同语义差别明显的词, 我们应该把系统判断的准确率定在 90% 左右, 反之, 对于语义模糊的情况, 如 *title*, *side* 和 *way*, 应该使用一个比较宽松的标准。

下界和底线(baseline)是可能的最简单算法的效能, 通常来说它是把所有上下文中的语义指定为使用最频繁的语义。这种算法性能的下界是必不可少的, 因为简单的准确度数字并不能真实说明某个词消歧处理的难度。比如, 对于一个带有两个相似语义的歧义词, 90% 的消歧正确率是一个很出色的结果, 而相同的正确率对于一个语义出现频率 9:1 的歧义词来说是微不足道的。通过总是选择最频繁语义的方法, 就可以得到这样的效果, 所以我们必须给出消歧算法的底线性能, 这样才可以判断所采用方法的优劣性。

▼ 当处理一个分类问题和评价方法准确度的时候, 会经常涉及到上界和下界。8.1 节讨论了其他的评价尺度, 具体来说就是精确率(precision)和召回率(recall)。

7.2 有监督消歧

在有监督消歧(Supervised Disambiguation)中, 一个已经消歧的语料库被用来训练。在这个样本训练集中, 歧义词 w 每一次出现都被标注上了一个语义标签(通常是符合上下文的语义 s_k)。这样就可以为有监督消歧提供一个统计分类的实例, 关于统计分类的讨论见第 16 章。统计分类的任务就是构建一个分类器, 根据上下文 c_i 对新的歧义词进行分类。另外, 表 7.1 中列出了本章将要使用的一些符号。

表 7.1 本章中使用的符号约定

符号	表示含义
w	一个歧义词
$s_1, \dots, s_k, \dots, s_K$	歧义词 w 的语义项
$c_1, \dots, c_i, \dots, c_l$	w 在语料库中的上下文
$v_1, \dots, v_j, \dots, v_J$	消歧使用的上下文特征(词)

在众多的有监督学习算法中, 我们选择了两个已经应用到语义消歧中的算法, 它们代表了统计自然语言处理中的两个重要理论方法: 贝叶斯分类(这种方法是(Gale et al. 1992b)提出的)和信息论(这种方法是 Brown et al. (1991b)提出的)。这两种方法也证明了完全不同的信息源是可以应用到消歧算法中的。第一种方法把上下文看做是一个无结构词集, 整合了上下文窗口中众多的词汇信息。第二种方法仅仅考虑了上下文中的一个信息特征, 这个信息特征可以很灵敏地反映上下文的结构。但是, 这个特征要谨慎地从大量的潜在信息中选取。

^① 参见(Jorgensen 1990)。为了能够正确比较不同处理任务之间相互判断的准确性的程度, 我们需要校正期望机会的一致性(expected chance agreement), 这取决于可区别语义的数目。这一点可以通过kappa统计学来实现(Siegel and Castellan 1988; Carletta 1996)。

7.2.1 贝叶斯分类

语义贝叶斯分类器的原理是,它在一个大的上下文窗口中考虑了歧义词周围的词的信息。每个实词都含有潜在的有用信息,暗示歧义词的哪个语义被使用。这个分类器不是进行特征选择,而是组合了所有特征。我们的形式化描述来自于 Gale et al. (1992b)。分类器的有监督训练要求语料库中的歧义词都事先被正确的语义标注。

在选择类别的时候,贝叶斯分类器使用贝叶斯决策规则,这个规则最小化了错误概率(Duda and Hart 1973: 10~43):

贝叶斯决策规则

$$\text{Decide } s' \text{ if } P(s'|c) > P(s_k|c) \text{ for } s_k \neq s' \quad (7.2)$$

贝叶斯决策规则是最优化的,因为它最小化了错误概率。对于每个独立的例子,它选择带有最高条件概率的类(或语义),因此有最小的错误率。对于一系列类决策(例如,在一个多页文本中 w 的所有消歧例子)的错误率也会尽可能小。

通常 $P(s_k|c)$ 的值是未知的,但是可以使用 2.1.10 节中的贝叶斯规则来计算它:

$$P(s_k|c) = \frac{P(c|s_k)}{P(c)} P(s_k)$$

$P(s_k)$ 是语义 s_k 的先验概率,它是不知道任何上下文信息时得到的 s_k 出现的概率。 $P(s_k)$ 被综合了上下文信息的因子 $P(c|s_k)/P(c)$ 校正,然后得到后验概率 $P(s_k|c)$ 。

如果仅仅是要得到正确的分类,那么可以通过消除 $P(c)$ (对于所有的语义,它是一个常量,因此不会影响最大值的选择)来简化这个分类任务。也可以使用概率的对数值来简化计算。然后,我们就能为 w 指定一个语义 s' :

$$\begin{aligned} s' &= \arg \max_{s_k} P(s_k|c) \\ &= \arg \max_{s_k} \frac{P(c|s_k)}{P(c)} P(s_k) \\ &= \arg \max_{s_k} P(c|s_k) P(s_k) \\ &= \arg \max_{s_k} [\log P(c|s_k) + \log P(s_k)] \end{aligned} \quad (7.3)$$

Gale 等人的分类器是一个特殊的贝叶斯分类器,即单纯贝叶斯分类器(Naive Bayes classifier)。由于效率很高并且可以混合大量的特征参数,单纯贝叶斯被广泛地应用到机器学习中(Mitchell 1997: ch.6)。如果我们把分类所基于的状态空间描述成一系列的特征,那么这个贝叶斯分类器就是可用的。在这种情况下,根据出现在上下文中的词 v_j 来描述词 w 的上下文。

单纯贝叶斯假设(Naive Bayes assumption)是指用来刻画事物特征的属性都是条件独立的:

单纯贝叶斯假设

$$P(c|s_k) = P(\{v_j|v_j \text{ in } c\}|s_k) = \prod_{v_j \text{ in } c} P(v_j|s_k) \quad (7.4)$$

在语义消歧中,单纯贝叶斯假设有两个推论。第一个是上下文中的所有结构和词语顺序

都可以被忽略。这通常是指一个可有重复的单词集模型 (bag of words model)^①。另一个推论是指在可有重复的单词集中出现的词独立于其他词。这两个推论显然是不对的。例如, *president* 出现在包含 *election* 的上下文中, 要比出现在包含 *poet* 的上下文中的可能性大很多。但是, 在很多情况下, 简化事件复杂度的假设使系统采用一个好的模型成为可能。虽然这个模型会有缺点, 但是它非常有效。很明显, 如果属性之间有很强的条件依赖关系, 那么单纯贝叶斯假设是不合适的。但是, 令人惊异的是, 在很多情况下单纯贝叶斯分类器的效果很好, 其部分原因是, 即使由于特征依赖性带来了概率估计的不准确, 分类器的决策仍然可以是最优的 (Domingos and Pazzani 1997)。

根据单纯贝叶斯假设, 我们在下面给出了修正的分类决策规则:

单纯贝叶斯决策规则

$$\text{Decide } s' \text{ if } s' = \arg \max_{s_k} [\log P(s_k) + \sum_{v_j \in c} \log P(v_j | s_k)] \quad (7.5)$$

公式中的 $P(v_j | s_k)$ 和 $P(s_k)$ 可以使用最大似然估计从标注的训练语料中计算得出 (最好带有合适的平滑方法):

$$P(v_j | s_k) = \frac{C(v_j, s_k)}{\sum_t C(v_t, s_k)}$$

$$P(s_k) = \frac{C(s_k)}{C(w)}$$

公式中 $C(v_j, s_k)$ 是训练语料中 v_j 在语义 s_k 的上下文中出现的次数, $C(s_k)$ 是 s_k 在训练语料中出现的次数, $C(w)$ 是歧义词 w 出现的总次数。图 7.1 中概述了这个算法。

```

1  comment: Training
2  for all senses  $s_k$  of  $w$  do
3      for all words  $v_j$  in the vocabulary do
4           $P(v_j | s_k) = \frac{C(v_j, s_k)}{\sum_t C(v_t, s_k)}$ 
5      end
6  end
7  for all senses  $s_k$  of  $w$  do
8       $P(s_k) = \frac{C(s_k)}{C(w)}$ 
9  end
10 comment: Disambiguation
11 for all senses  $s_k$  of  $w$  do
12      $\text{score}(s_k) = \log P(s_k)$ 
13     for all words  $v_j$  in the context window  $c$  do
14          $\text{score}(s_k) = \text{score}(s_k) + \log P(v_j | s_k)$ 
15     end
16 end
17 choose  $s' = \arg \max_{s_k} \text{score}(s_k)$ 

```

图 7.1 贝叶斯消歧

① 一个可有重复的单词集就像一个数学上的集合, 但是它允许有重复的元素。我们在式 (7.4) 中使用 “in” 代替 “ $\in$ ” 就是因为处理的 c 是一个可有重复的单词集。

Gale, Church and Yarowsky(1992b; 1992c)中报告,一个基于这种算法的消歧系统对 Hansard 语料库中的 6 个歧义名词的消歧正确率可以达到大约 90%,这 6 个名词是: *duty*, *drug*, *land*, *language*, *position* 和 *sentence*。

表 7.2 给出了一些例子,它们对 Hansard 语料库中 *drug* 的两个语义有很好的指示作用。例如, *prices* 对于语义 *medication* 是一个很好的线索。这就意味着 $P(\text{prices} | \text{'medication'})$ 比较大而 $P(\text{prices} | \text{'illegal substance'})$ 比较小,并且因此 *drug* 的上下文(包含有 *prices*)对于语义 *medication* 会有比较高的分数,而对语义 *illegal substance* 有比较低的分数(正如图 7.1 中第 14 行的计算)。

表 7.2 贝叶斯分类器使用的 *drug* 的两个语义线索。改编自 (Gale et al. 1992b:419)

语义	语义线索
medication	<i>prices, prescription, patent, increase, consumer, pharmaceutical</i>
illegal substance	<i>abuse, paraphernalia, illicit, alcohol, cocaine, traffickers</i>

7.2.2 一个基于信息论的方法

贝叶斯分类器试图使用上下文窗口中所有的词的信息来帮助进行消歧决策,而且还事先做出了一个不太真实的独立性假设。我们现在要讨论的信息论方法使用了相反的策略,试图寻找一个单一的上下文特征,它可以可靠地指示出歧义词的哪一种语义被使用。Brown et al. (1991b)的一些关于法语歧义词语义指示器的例子如表 7.3 所示。对于动词 *prendre*, 它的宾语是判断其语义的很好的指示器: *prendre une mesure* 翻译成 *to take a measure*, *prendre une décision* 翻译成 *to make a decision*。类似地,动词 *vouloir* 的时态和 *cent* 左边的词都是这两个词的语义的很好的指示器,如表 7.3 所示。

表 7.3 三个法语歧义词的充分信息指示器

歧义词	指示器	例子:值→语义
<i>prendre</i>	宾语	<i>measure</i> → <i>to take</i>
		<i>décision</i> → <i>to make</i>
<i>vouloir</i>	时态	<i>present</i> → <i>to want</i>
		<i>conditional</i> → <i>to like</i>
<i>cent</i>	左边的词	<i>per</i> →%
		<i>number</i> → <i>c. [money]</i>

为了更好地应用语料信息,信息的量值需要进行规范化,以指示语义,例如, *measure* 指示 *to take*, *décision* 指示 *to make*。Brown 等人使用了 Flip-Flop 算法来解决这个问题。设 $t_1, \dots, t_m$ 是歧义词的翻译(或语义), $x_1, \dots, x_n$ 是指示器可能的取值。图 7.2 给出了这种情况下的 Flip-Flop 算法。在这里描述的算法版本只能处理两个语义的歧义情况。参见 Brown et al. (1991a) 的处理两个以上语义的扩展版本。回忆一下 2.2.3 节中互信息的定义:

$$I(X; Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$$

我们可以看到,Flip-Flop 算法的每一次迭代都必须满足使互信息 $I(P; Q)$ 单调增加,所以算法的一个很自然的终止条件就是互信息 $I(P; Q)$ 不再增加或者增加很少。

例如,假设我们想要根据宾语翻译 *prendre*, 并且我们已经有了 $\{t_1, \dots, t_m\} = \{take, make,$

rise, speak 和 $\{x_1, \dots, x_n\} = \{measure, note, exemple, décision, parole\}$, 参见 (Brown et al. 1991b: 267)。最初的语义划分 P 可能是 $P_1 = \{take, rise\}$ 和 $P_2 = \{make, speak\}$ 。指示器 Q 的哪一种划分会得到最大的 $I(P; Q)$? 显然, 这要依据于我们处理的具体数据。那么假设当出现的宾语是 *measure, note* 和 *exemple* 时, *prendre* 会被翻译成 *take* (相应的短语是 *take a measure, take notes* 和 *take an exemple*), 当宾语是 *décision* 和 *parole* 时翻译成 *make, speak* 和 *rise* (相应的短语是 *make a decision, make a speech* 和 *rise to speak*)。

```

1 find random partition  $P = \{P_1, P_2\}$  of  $\{t_1, \dots, t_m\}$ 
2 while (improving) do
3     find partition  $Q = \{Q_1, Q_2\}$  of  $\{x_1, \dots, x_n\}$ 
4     that maximizes  $I(P; Q)$ 
5     find partition  $P = \{P_1, P_2\}$  of  $\{t_1, \dots, t_m\}$ 
6     that maximizes  $I(P; Q)$ 
7
8 end

```

图 7.2 发现消歧指示器的 Flip-Flop 算法

最大化 $I(P; Q)$ 的划分是 $Q_1 = \{measure, note, exemple\}$ 和 $Q_2 = \{décision, parole\}$, 因为这样可以给我们最大的信息, 以区分 P_1 和 P_2 中的翻译。我们只要在 *prendre la parole* 被翻译成 *rise to speak* 时加以纠正就可以了, 而且这样的错误是不可避免的, 因为 *rise* 和 *speak* 在两个不同的划分中。

算法接下来的两步是, 重新划分 P 为 $P_1 = \{take\}$ 和 $P_2 = \{make, rise, speak\}$, 并且 Q 还是以前的划分。这个划分总是在校正 *take*。如果我们想要区分其他的翻译, 如 *make, rise* 和 *speak*, 就不得不考虑多于两个语义的情况。

穷尽搜索法语翻译和最可能指示器的最佳划分会使搜索时间以指数递增。对于计算一个特殊指示器值的最佳划分, Flip-Flop 算法是一个有效的线性时间算法, 它基于分裂理论 (splitting theorem) (Breiman et al. 1984)。我们对所有可能的指示器使用这个算法, 然后选择带有最高互信息的指示器。Brown 等人发现这些最好的指示器是 *prendre* 的宾语, *vouloir* 的时态和 *cent* 左边的词, 如表 7.3 所示。

当指示器和它取值的特殊划分已经确定以后, 消歧就很简单了, 如下所示:

1. 对于出现的一个歧义词, 确定它的指示器值 x_i 。
2. 如果 x_i 在 Q_1 中, 指定歧义词的语义为语义 1; 如果 x_i 在 Q_2 中, 指定歧义词的语义为语义 2。

Brown et al. (1991b) 中报告, 当把信息论算法结合到机器翻译系统中时, 系统性能可以提高 20% (在 100 个句子中正确的句子由 37 个增加到 45 个)。

我们称这种算法为有监督的, 是因为它需要一个标注训练集。然而在 Brown et al. (1991b) 中, 每一个出现的例子, 比如法语 *cent*, 并没有标注其语义, 而是用它对应的英语翻译来标注。这些类别标签 (label) 不是语义。例如, 法语 *cent* 的一些标签是 (英语) *per* 和数字 0, *one*, 2 和 8。算法把这些标签分为两类: $Q_1 = \{per\}$ 和 $Q_2 = \{0, one, 2, 8\}$, 它们可以被解释为 *cent* 的两个语义, 相应于英语翻译 % 和 *cent*。因此这里存在一个标签到语义的多对一映射。

7.3 基于词典的消歧

如果一个词没有语义范畴信息,我们可以求助于它的一般语义描述。本节讨论的消歧方法就是依赖于词典和类义辞典中的语义定义的。在这里我们使用了3种不同的信息类型。Lesk(1986)直接使用词典中的语义定义。Yarowsky(1992)提出了怎样把词的语义范畴(从 Roget 类义辞典中的范畴得到)应用到上下文的语义范畴和消歧判断中。在 Dagan and Itai(1994)的方法中则是从一个双语词典得到不同语义的翻译,通过分析它们在一个对应外语语料中的分布来进行消歧。最后我们将会看到,考察语义的分布属性可以大大提高消歧效果。任意一篇给定的文本和一个搭配中的歧义词通常都只使用一个语义(假设每个搭配和每篇文本都只有一个语义)。

7.3.1 基于语义定义的消歧

Lesk(1986)提出了一个新的消歧思路,认为词典中词条本身的定义就可以作为判断其语义的一个很好的依据条件^①。假设词典中 *cone* 的两个定义如下所示:

1. a mass of ovule-bearing or pollen-bearing scales or bracts in tree of the pine family or in cycads that are arranged usually on a somewhat elongated axis,
2. something that resembles a cone in shape: as ... a crisp cone-shaped wafer for holding ice cream.

如果 *tree* 或者 *ice* 和 *cone* 出现在相同的上下文中,那么可能说明 *cone* 的语义就是那个包含着该词的词典定义语义:*tree* 对应语义1,*ice* 对应语义2。

设 $D_1, \dots, D_k$ 是歧义词 w 的语义 $s_1, \dots, s_k$ 的词典定义,在定义中它们被看成是一个可有重复的单词集, E_{v_j} 表示在 w 的上下文中出现的词 v_j 的词典定义,在 v_j 的定义中它也被表示为一个可有重复的单词集。如果 v_j 的语义是 $s_{j1}, \dots, s_{j\ell}$, 那么 $E_{v_j} = \bigcup_{\mu} D_{\mu}$ 。为了简单起见,我们忽略了在 w 的上下文中词 v_j 的语义区分。Lesk 的算法的具体过程描述如图 7.3 所示。对于算法中的 *overlap* 计算函数,我们只是统计在语义 s_k 的定义 D_k 和 $\bigcup_{v_j \text{ in } c} E_{v_j}$ 集合中同现的词数目(c 表示 w 上下文中出现的单词)。或者我们也可以使用在表 8.7 中列出的任意类似计算方法。

```

1 comment: Given: context  $c$ 
2 for all senses  $s_k$  of  $w$  do
3    $\text{score}(s_k) = \text{overlap}(D_k, \bigcup_{v_j \text{ in } c} E_{v_j})$ 
4 end
5 choose  $s'$  s.t.  $s' = \arg \max_{s_k} \text{score}(s_k)$ 

```

图 7.3 Lesk 的基于词典的消歧算法。 D_k 是语义 s_k 的词典定义。 E_{v_j} 是词 v_j 的词典定义中出现的词集(换句话说,就是所有 v_j 语义定义的联合)

作为 Lesk 算法的例子, *ash* 的语义示于表 7.4 中。表 7.5 中的上下文可以通过统计不同语义定义中同现词的数目来正确地区分歧义。

^① Lesk 把算法的最初思路归功于 Margaret Millar 和 Laurence Urdang。

表 7.4 ash 的两个语义

语义	定义
s_1 tree	a tree of the olive family
s_2 burned stuff	the solid residue left when combustible material is burned

表 7.5 使用 Lesk 算法对 ash 进行语义消歧。分值表示语义定义和上下文共享词的数目。第一个句子中 ash 被消歧的语义是 burned stuff, 因为有一个上下文词 burn 和语义 2 的定义是共享的, 而且没有词和其他语义共享。在第二个例子中, 和语义 1 (tree) 共享的词是 tree

语义	上下文
s_1 s_2	
0 1	This cigar burns slowly and creates a stiff ash.
1 0	The ash is one of the last trees to come into leaf.

单独来看, 从词典得到的这些信息对于高质量的语义消歧是不够的。Lesk 对一个歧义词样本使用这个算法时, 报告的准确率只能在 50% ~ 70%。但同时他提出了各种优化方法来提高系统效能。例如, 可以在文本中迭代几次算法。我们可以不使用 v_j 定义中出现的所有词 E_{v_j} 的联合, 而仅仅使用上一次算法迭代已经确定的语义定义。有人会有这样的想法, 希望算法迭代最终得到文本中所有词的正确语义。Pook and Catlett (1988) 提出了另一种改进方法: 通过类义辞典中的同义词列表来扩展上下文中的每个词。这样的算法混合了基于词典和基于类义辞典消歧的特性。

7.3.2 基于类义辞典的消歧

基于类义辞典的消歧方法使用了语义范畴 (semantic categorization) 信息, 这些语义范畴信息是由类义辞典 (如 Roget (Roget 1946) 的类义辞典) 或者类似 Longman (Procter 1978) 的带有主题范畴 (subject categories) 的词典提供的。基于类义辞典消歧的基本原理, 上下文中词汇的语义范畴大体上确定了这个上下文的语义范畴, 并且上下文的语义范畴可以反过来确定词汇的哪一个语义被使用。

下面给出了 Walker (1987: 254) 提出的一个基于类义辞典的简单消歧算法。这个算法使用的基本信息是词典中给每个词指定的一个或多个主题码 (subject code)。如果一个词被指定了多个主题码, 那么假定它们就相应于这个词的不同语义。设 $t(s_k)$ 为上下文中歧义词 w 的语义 s_k 的主题码。那么, 通过统计类义辞典中可以把 $t(s_k)$ 列为可能主题的上下文词的个数, 就可以对 w 进行消歧处理。我们选择带有最高统计值的语义作为消歧的结果, 算法如图 7.4 所示。

```

1 comment: Given: context  $c$ 
2 for all senses  $s_k$  of  $w$  do
3    $\text{score}(s_k) = \sum_{v_j \in c} \delta(t(s_k), v_j)$ 
4 end
5 choose  $s'$  s.t.  $s' = \arg \max_{s_k} \text{score}(s_k)$ 

```

图 7.4 基于类义辞典的消歧算法。如果 $t(s_k)$ 是 v_j 的主题码之一, 那么 $t(s_k)$ 为语义 s_k 的主题码, $\delta(t(s_k), v_j) = 1$; 在其他情况下 $\delta(t(s_k), v_j) = 0$ 。score 是具有语义 s_k 主题码的词数目

Black(1988: 187)在5个歧义词样本上应用 Walker 的算法,仅取得了一定程度上的成功:准确率大约在 50%左右。但是,我们注意到他选取的测试词都具有很强的歧义性: *interest*, *point*, *power*, *state* 和 *terms*。

这种算法的一个问题是,一般的主题范畴并不适合于特殊领域。例如,在类义辞典中 *mouse* 有两个语义范畴:哺乳动物(mammal)和电子器件(electronic device),但是在计算机手册中很少会出现范畴“哺乳动物”。一般的主题领域也会有覆盖面的问题。例如,我们不会在 20 世纪 60 年代的任何一本类义辞典中找到 *Navratilova* (娜弗拉蒂诺娃)这个词(而且也不会找到任何相关的名词)。但是 *Navratilova* 的出现对于范畴“sports”是一个很好的指示。

图 7.5 中给出的算法增加了对语料库的主题分类,是由 Yarowsky(1992)设计的。这个算法把语料库中 t_i 类上下文中频繁出现的词加到了 t_i 类里。例如, *Navratilova* 在 sports 类文本里的出现次数要远远大于其他类文本,所以可以把它加到 sports 类中。

```

1 comment: Categorize contexts based on categorization of words
2 for all contexts  $c_i$  in the corpus do
3   for all thesaurus categories  $t_l$  do
4      $\text{score}(c_i, t_l) = \log \frac{P(c_i|t_l)}{P(c_i)} P(t_l)$ 
5   end
6 end
7  $t(c_i) = \{t_l | \text{score}(c_i, t_l) > \alpha\}$ 
8 comment: Categorize words based on categorization of contexts
9 for all words  $v_j$  in the vocabulary do
10    $V_j = \{c | v_j \text{ in } c\}$ 
11 end
12 for all topics  $t_l$  do
13    $T_l = \{c | t_l \in t(c)\}$ 
14 end
15 for all words  $v_j$ , all topics  $t_l$  do
16    $P(v_j|t_l) = |V_j \cap T_l| / \sum_j |V_j \cap T_l|$ 
17 end
18 for all topics  $t_l$  do
19    $P(t_l) = (\sum_j |V_j \cap T_l|) / (\sum_l \sum_j |V_j \cap T_l|)$ 
20 end
21 comment: Disambiguation
22 for all senses  $s_k$  of  $w$  occurring in  $c$  do
23    $\text{score}(s_k) = \log P(t(s_k)) + \sum_{v_j \text{ in } c} \log P(v_j|t(s_k))$ 
24 end
25 choose  $s'$  s.t.  $s' = \arg \max_{s_k} \text{score}(s_k)$ 

```

图 7.5 调整的基于类义辞典消歧的 Yarowsky 算法。该算法基于类义辞典消歧并调整词汇的语义类。第16行中的 $P(v_j|t_l)$ 是含有词 v_j 的 topic t_l 的上下文所占比值的估计

Yarowsky 提出了一个基于 7.2.1 节中介绍的贝叶斯分类器的变种算法来完成消歧处理。首先我们计算语料库中上下文 c_i 和类义辞典中义类 t_l 的分值。例如,上下文(7.6)对于类义辞典的 sports 类会有很高的分数,假设类义辞典已经把 *tennis* 列为一个 sports 类的词。在 Yarowsky 的实验中,一个上下文就是以歧义词为中心的 100 个词的窗口。

It is amazing that Navratilova, who turned 33 earlier this year, continues to play great tennis.

(7.6)

根据单纯贝叶斯假设,我们可以计算 $\text{score}(c_i, t_i)$ 为 $\log P(t_i | c_i)$, 在这里 $P(t_i | c_i)$ 可以根据以下公式计算:

$$\begin{aligned} P(t_i | c_i) &= \frac{P(c_i | t_i)}{P(c_i)} P(t_i) \\ &= \frac{\prod_{v \in c_i} P(v | t_i)}{\prod_{v \in c_i} P(v)} P(t_i) \end{aligned} \quad (7.7)$$

然后使用算法的第 7 行中的阈值 α 来确定哪一类义辞典范畴最适合这个上下文。阈值通常选择非常大的值,因此只有和这个范畴最适合的上下文会被指定。

现在我们可以根据语料库(表示为上下文 $\{c_i\}$ 的集合)来调整类义辞典中的语义类。在算法的第 16 行中,我们用属于类 t_i 的 v_j 的上下文所占的比例来估计 $P(v_j | t_i)$ 。如果类义辞典中已经有了 v_j ,那么算法需要根据语料库来调整 v_j 的语义类(例如, *stylus* 作为计算机术语可以有很高的分数,但是在类义辞典中它的语义类仅仅是 *writing*)。如果 v_j 在辞典类义中没有,那么算法会把它加到一个合适的语义类中(就像 *Navratilova* 的例子)。 t_i 的先验概率简单地用相对频率来计算,用来调整有些上下文没有语义类而有些上下文有多于一个语义类的情况(见第 19 行)。

使用第 16 行计算出的 $P(v_j | t_i)$ 就可以进行消歧,方法和我们先前介绍的贝叶斯算法类似(参见图 7.1)。(Yarowsky 1992)推荐了一种最大似然估计的平滑方法(参见第 6 章)。

表 7.6 给出了一些(Yarowsky 1992)的结果。当类义辞典中的范畴和语义与主题很好地吻合时(比如 *bass* 和 *star*),这种方法有很高的准确率。但是当语义涉及到几个主题时,算法效果通常很差。Yarowsky 把这种情况称之为语义之间的主题独立性(topic-independent distinctions)。例如, *interest* 的 *advantage* 语义(如 *self-interest*)不是一个特殊的主题。*self-interest* 可以出现在音乐、娱乐、空间探索和金融等多种领域。因此,基于主题的分类法在这个语义上效果很差。

表 7.6 基于词典消歧的一些结果。这个表显示了三个歧义词的语义,它们相应的 Roget 范畴和图 7.5 中算法的准确率。改编自 (Yarowsky 1992)

单词	语义	Roget 范畴	准确率
<i>bass</i>	musical senses	MUSIC	99%
	fish	ANIMAL, INSECT	100%
<i>star</i>	space object	UNIVERSE	96%
	celebrity	ENTERTAINER	95%
	star shaped object	INSIGNIA	82%
<i>interest</i>	curiosity	REASONING	88%
	advantage	INJUSTICE	34%
	financial	DEBT	90%
	share	PROPERTY	38%

7.3.3 在第二语言语料库翻译基础上的消歧

第三种基于词典的算法使用了双语词典中的对应词(Dagan et al. 1991; Dagan and Itai

1994)。我们把使用语言(即需要消歧的语言)称为第一语言,并且把在双语词典中的目标语言称为第二语言。例如,如果我们基于一个德语语料库对英语进行消歧,那么英语是第一语言,德语是第二语言,并且我们需要一个英德双语词典。

表 7.7 给出的例子很好地阐述了 Dagan 和 Itai 算法的基本思想。英语 *interest* 有两个语义并在德语中有两个不同的翻译。语义 1 翻译为 *Beteiligung* (*legal share*, 例如 *a 50% interest in the company*), 语义 2 翻译为 *Interesse* (*attention, concern*, 例如 *her interest in mathematics*)。我们在这里忽略了 *interest* 的其他语义。为了对英语中的 *interest* 进行消歧,我们识别有 *interest* 出现的短语,并且使用这个短语搜索一个德语语料库中的实例。如果这个短语只和 *interest* 的德语翻译中的一个共现,那么我们就可以对使用 *interest* 的短语指定相应的语义。

表 7.7 如何使用第二语言语料库对 *interest* 消歧

	语义1	语义2
定义	legal share	attention, concern
德语翻译	<i>Beteiligung</i>	<i>Interesse</i>
搭配	<i>acquire an interest</i>	<i>show interest</i>
德语翻译	<i>Beteiligung erwerben</i>	<i>Interesse zeigen</i>

举一个例子,假设 *interest* 出现在词组 *showed interest* 中。而 *show* 的德语翻译 *zeigen* 只和 *Interesse* 一起出现,因为语义 *legal shares* 一般不说显示(*show*)。所以可以得出结论,在词组 *to show interest* 中出现的 *interest* 的语义是 *attention, concern*。另一方面,词组 *acquired an interest* 频繁出现的翻译是 *erwarb eine Beteiligung*, 因为语义 *attention, concern* 一般不说需要(*acquire*)。这就告诉我们 *interest* 作为 *acquire* 的宾语使用时,相应的语义是 *legal share*。

图 7.6 给出了这个算法的一个简单实现。对于上面的例子,关系 *R* 是 *is-object-of*, 并且我们的目的是使用关系 *R*(*interest, show*)中对 *interest* 消歧。为了做到这一点,需要在第二语言语料库中统计 *interest* 的两种语义的翻译和 *show* 的翻译同现的次数。我们发现 *R*(*Interesse, zeigen*)的次数高于 *R*(*Beteiligung, zeigen*)的次数,所以选择对应于 *Interesse* 的语义 *attention, concern* 作为结果。

```

1 comment: Given: a context c in which w occurs in relation R(w, v)
2 for all senses sk of w do
3   score(sk) = |{c ∈ S | ∃ w' ∈ T(sk), v' ∈ T(v) : R(w', v') ∈ c}|
4 end
5 choose s' = arg maxsk score(sk)

```

图 7.6 基于第二语言语料库的消歧算法。*S* 是第二语言语料库, *T*(*s_k*) 是语义 *s_k* 可能的翻译集, *T*(*v*) 是 *v* 可能的翻译集。一个语义的 *score* 是它的翻译之一和 *v* 的翻译在第二语言语料库中同现的次数

Dagan 和 Itai 使用的算法更复杂一些:只有在决策可以信赖的时候,才进行消歧。考虑希伯来语 *ro'sh* 的例子,它可以有两个可能的英语翻译: *top* 和 *head*。Dagan 和 Itai 在他们的英语第二语言语料库中找到了 10 个 *stand at head* 例子和 5 个 *stand at top* 例子。这表明 *stand at head* 有很大的可能是希伯来短语 *amad be-ro'sh* 的翻译。但是,预期 *stand at head* 是错误翻译的可能性也很大(大约有 $5/(5+10) \approx 0.33$)。在很多情况下,不做结论比得到一个错误概率较高的结论更好。例如,在一个大系统中,很多组成部分都有一定的错误率,所以上面的例子中

大约 0.67 的正确率是不能被接受的。如果一个句子被分成了 5 个部分,每个部分的错误率是 0.33,那么整个系统的正确率就只有 $14\% : (1 - 0.33)^5 \approx 0.14$ 。Dagan 和 Itai 说明了如何估计错误概率。只有当置信度大于等于 90% 时才会给出决定。

7.3.4 每篇文本一个语义,每个搭配一个语义

迄今为止我们考虑的基于词典的算法都是分别处理每个出现的歧义词。但是歧义词之间还有一些限制。本节讨论的内容就是 Yarowsky (1995) 提出的这样两个约束。

- 每篇文本一个语义。在任意给定的文本中目标词语义有很强的一致性。
- 每个搭配一个语义。根据和目标词之间的相对距离、次序和句法关系,相邻词提供了可用于判断目标词语义的很多线索信息。

对于第一个限制,我们考虑例子 *plant*。这个约束表明,如果 *plant* 的第一次出现使用的语义是 *living being*,那么它后面的出现也非常有可能使用语义 *living being*。表 7.8 给出了两个例子。特别是当歧义词在小文本中,或者是在一个可以分割为小文本的语料中时(使用 15.5 节讨论的方法分割文本),这个约束一般非常有用。在下面将会看到,语义的这个简单特征可以被有效地使用到消歧中。

表 7.8 每篇文本一个语义约束的例子。表显示两个不同文本 d_1 和 d_2 的上下文。在 d_1 的一个上下文中缺少足够的局部信息来消除歧义(“?”)。在 d_2 的最后一个上下文中局部信息误导了消歧。每篇文本一个语义的约束可以抵消信息不足和误导信息带来的影响。它可以正确地无类别的和缺少类别的上下文指定语义 *living*。改编自 Yarowsky (1995)

文本	初始标号	上下文
d_1	living	the existence of <i>plant</i> and animal life
	living	classified as either <i>plant</i> or animal
	?	Although bacterial and <i>plant</i> cells are enclosed
d_2	living	contains a varied <i>plant</i> and animal life
	living	the most common <i>plant</i> life
	living	slight within Arctic <i>plant</i> species
	factory	are protected by <i>plant</i> parts remaining from

第二个约束做了一个基本假设,大部分统计消歧研究都要依赖的条件是:语义和某些上下文特征有很强的联系,比如相同短语中的其他词就是一个上下文特征。Yarowsky (1995) 的方法和 Brown et al. (1991b) 的信息论方法是类似的,关于信息论的方法我们已经在 7.2.2 节中介绍过了。在 Yarowsky 的方法中,他选择了一个很强的搭配特征并且仅在这个特征的基础上进行消歧。搭配特征可以进行排列:

$$\frac{P(s_{k_1}|f)}{P(s_{k_2}|f)} \quad (7.8)$$

这个公式本质上是语义 s_{k_1} 和搭配 f 的同现次数除以语义 s_{k_2} 和搭配 f 同现次数的比值。此外,如果搭配和语义出现得不频繁,需要进行平滑处理,参见 Yarowsky (1994)。

只根据一个强特征进行消歧有这样的优点:无需对不同特征进行混合。很多统计方法,比如在 7.2.1 节中使用的单纯贝叶斯方法和本节稍前部分讨论的基于词典的方法,在混合特征

的时候都必须假设特征具有独立性。由于这种独立性假设是很难成立的,所以有时需要避免进行特征混合,而仅仅根据一个可信度比较高的特征进行估计,这样做的算法效果会比较好。此外还有一个更复杂的可供选择的方法,即对已知信息源(sources of evidence)之间的依赖关系建立准确的模型,具体的内容参见第16章。

Yarowsky 混合这两种约束设计了一个算法,图 7.7 是这个算法的示意性描述。算法对每一个语义 s_k 都迭代建立了两个独立集。 F_k 包含特征搭配。 E_k 是歧义词 w (当前被指定为语义 s_k) 的上下文。

```

1  comment: Initialization
2  for all senses  $s_k$  of  $w$  do
3       $F_k$  = the set of collocations in  $s_k$ 's dictionary definition
4  end
5  for all senses  $s_k$  of  $w$  do
6       $E_k = \emptyset$ 
7  end
8  comment: One sense per collocation
9  while (at least one  $E_k$  changed in the last iteration) do
10     for all senses  $s_k$  of  $w$  do
11          $E_k = \{c_i | \exists f_m : f_m \in c_i \wedge f_m \in F_k\}$ 
12     end
13     for all senses  $s_k$  of  $w$  do
14          $F_k = \{f_m | \forall n \neq k \frac{P(s_k|f_m)}{P(s_n|f_m)} > \alpha\}$ 
15     end
16 end
17 comment: One sense per discourse
18 for all documents  $d_m$  do
19     determine the majority sense  $s_k$  of  $w$  in  $d_m$ 
20     assign all occurrences of  $w$  in  $d_m$  to  $s_k$ 
21 end

```

图 7.7 基于“每篇文本一个语义”和“每个搭配一个语义”的消歧算法

在第3行中, F_k 初始为 s_k 的词典定义或者其他来源(例如,词典编撰者手工加入的搭配集或者是从小规模的手工标注训练语料中得到搭配集)。 E_k 初始为空集。

迭代开始的时候,把 F_k 中一个特征搭配的所有上下文赋给 E_k (第11行)。例如,如果“是 show 的宾语”是 $F_{\text{attention, concern}}$ 中的一个搭配,那么 *interest* 的上下文(在这个上下文中 *interest* 是 *show* 的宾语)会被赋给 $E_{\text{attention, concern}}$ 。然后重新计算特征搭配集,选择的搭配要在刚刚更新的 E_k 中排列靠前(第14行)。

在算法的这部分完成以后,已经使用了“每篇文本一个语义”约束。在文档或文本中的歧义词的语义被指定为一个主要的语义(第20行)。表 7.8 给出的两个例子就是这个算法的结果。

Yarowsky 证明这个算法是很有效的。不同版本得到的准确率在 90.6%到 96.5%之间。在合并文本约束(第18行至第21行)的时候错误率减少了 27%。令人惊异的是,不需要标注训练事例的算法给出了这么好的结果。

7.4 无监督消歧

上一节讨论的消歧方法都需要一些基本的词典资源、小训练集或者一些搭配种子集(seed)。虽然这些方法需要的信息量似乎很少,但是我们仍然会遇到一些情况,在这些情况中这些小的信息需求也是无法满足的。特别是当我们处理特殊领域信息时,会经常遇到这样的情况,在这些特殊领域中可能没有我们需要的词汇资源信息^①。例如,信息检索系统要处理各种主题领域的文本。一般的普通词典不太适用于领域特殊的文本。化学文摘数据库就主要包括化学范畴内的文档。因此,基于类义辞典的消歧算法的效果就会大打折扣。我们不能期望信息检索系统的用户来定义这些歧义词的语义,或者是对新文本集提供一个训练集。随着近年来在线资源的急速增加,给消歧带来了大量的难题,外部的信息源对于消歧的可用性并不大。

严格地讲,如果我们想要得到语义标注,完全无监督的消歧是不可能的,这里说的语义标注是一个可以标注词汇的具体语义算法。语义标注需要提供语义的一些特征描述。但是,语义辨别(sense discrimination)可以在完全无监督的形式下实现。我们可以把歧义词的上下文聚类到很多分组中,然后在这些组之间就可以无标记地辨别它们。几种这样的语义辨别算法已经被设计出来了。我们将会在这里描述它们中的一个:分组上下文判别(context-group discrimination),这个算法在很大程度上是根据 Schütze(1998)^② 改编的。我们还要注意这个方法和 7.2.2 节中描述的 Brown 等人的方法的相似性。Brown et al. (1991b) 对一个歧义词的翻译进行聚类,这可以看成是对一个歧义词 w 出现实例的预标注。在这里我们考虑的是完全的无监督算法,它聚类的文本是没有标注的。

这种方法的概率模型和 7.2.1 节中 Gale 等人的模型是相似的。对于一个具有语义 $s_1, \dots, s_k, \dots, s_K$ 的歧义词 w , 我们估计歧义词 w (使用语义 s_k) 的上下文中出现 v_j 的条件概率, 也就是 $P(v_j | s_k)$ 。

和 Gale 等人的贝叶斯分类器不同,在无监督消歧中的参数估计不是根据有标注的训练数据。相反,我们开始的时候随机初始化参数 $P(v_j | s_k)$, 然后根据 EM 算法(参见 14.2.2 节)重新估计 $P(v_j | s_k)$ 。在随机初始化之后,对 w 的每一个上下文 c_i 计算概率 $P(c_i | s_k)$ 。并且可以使用上下文的初始分类作为我们的训练数据,然后重新估计参数 $P(v_j | s_k)$,使得模型给定数据的似然值最大。这种算法如图 7.8 所示。

EM 算法要保证每一步中模型似然对数的值是增加的,因此这个算法的终止条件是似然值(在第一步计算的)不再明显地增加。

当评价了模型的参数之后,我们就可以通过计算每个语义的概率来消歧 w 的上下文,这些语义是出现在上下文中 v_j 的语义。此外,做一个单纯贝叶斯假设并使用贝叶斯判别规则(7.5):

$$\text{Decide } s' \text{ if } s' = \arg \max_{s_k} [\log P(s_k) + \sum_{v_j \in c} \log P(v_j | s_k)]$$

① 但是在一些领域中有专门的词典,比如医学和科学术语词典。

② 出于一致性考虑,我们使用 7.2.1 节和 7.3.2 节中使用的概率模型来代替 Schütze 的概率模型。

1. **Initialize** the parameters of the model μ randomly. The parameters are $P(v_j|s_k)$, $1 \leq j \leq J$, $1 \leq k \leq K$, and $P(s_k)$, $1 \leq k \leq K$.

Compute the log of the likelihood of the corpus C given the model μ as the product of the probabilities $P(c_i)$ of the individual contexts c_i (where $P(c_i) = \sum_{k=1}^K P(c_i|s_k)P(s_k)$):

$$l(C|\mu) = \log \prod_{i=1}^I \sum_{k=1}^K P(c_i|s_k)P(s_k) = \sum_{i=1}^I \log \sum_{k=1}^K P(c_i|s_k)P(s_k)$$

2. While $l(C|\mu)$ is improving repeat:

- (a) **E-step.** For $1 \leq k \leq K$, $1 \leq i \leq I$ estimate h_{ik} , the posterior probability that s_k generated c_i , as follows:

$$h_{ik} = \frac{P(s_k)P(c_i|s_k)}{\sum_{k=1}^K P(s_k)P(c_i|s_k)}$$

To compute $P(c_i|s_k)$, we make the by now familiar Naive Bayes assumption:

$$P(c_i|s_k) = \prod_{v_j \in c_i} P(v_j|s_k)$$

- (b) **M-step.** Re-estimate the parameters $P(v_j|s_k)$ and $P(s_k)$ by way of maximum likelihood estimation:

$$P(v_j|s_k) = \frac{\sum_{\{c_i: v_j \in c_i\}} h_{ik}}{Z_k}$$

where $\sum_{\{c_i: v_j \in c_i\}}$ sums over all contexts in which v_j occurs and $Z_k = \sum_j \sum_{\{c_i: v_j \in c_i\}} h_{ik}$ is a normalizing constant.

Recompute the probabilities of the senses as follows:

$$P(s_k) = \frac{\sum_{i=1}^I h_{ik}}{\sum_{k=1}^K \sum_{i=1}^I h_{ik}} = \frac{\sum_{i=1}^I h_{ik}}{I}$$

图 7.8 语义聚类学习的 EM 算法。 K 是希望的语义数目; $c_1, \dots, c_i, \dots, c_I$ 是在语料库中歧义词的上下文; $v_1, \dots, v_j, \dots, v_J$ 是消歧的特征词

通过在一定的语义数目 K 内运行算法,我们可以选择歧义词语义分类的粒度。语义越多,模型的结果就越复杂,因此它也就能更好地反映数据的情况。结果,随着每个新语义的加入,模型似然对数值会越来越高。但是,我们可以考察每个新语义加入后模型似然值增加的幅度,以确定是否可以加入新语义。如果似然对数值增加幅度很明显,表明新语义反映了数据的一个主要部分,那么这个新的语义数目就是可取的。如果似然对数值只是一般性地增加,那么新语义仅仅是反映了数据的随机变化,表明它可能是不可取的。

一个确定语义数目的更简单方法是,在语义数目和可用训练数据的规模之间建立依赖关系, Schütze and Pedersen(1995)将这种方法应用于信息检索,结果证明该方法相当有效。

无监督消歧的一个优点是,它更适合区分有细微差别的语义用法,这些语义的细微差别在词典中找不到。此外,信息检索也可以应用无监督消歧。比如, *bank* 的语义在 *bank robberies* 的

上下文中是银行实体,在 *corporate mergers* 的上下文中是抽象团体,这两个语义之间的区别很小,甚至在词典中都没有反映出来。

如果无监督算法处理的语义数目很大,比如说 $K = 20$,那么算法会把词典语义划分到更细微的上下文变量中。例如, *suit* 的 *lawsuit* 语义就会被划分到 *civil suit*, *criminal suit* 等语义中。通常,归纳聚类(induced cluster)不会根据词典语义很好地排列。不频繁出现的语义和只有很少搭配的语义很难在无监督消歧中单独列出来。例如, *suit* 的语义 *to be appropriate for* (出现在 *suit me fine* 中)就不太可能被发现。但是,这些难于识别的语义所带的内容一般也没有特殊领域中的特殊语义重要。对于信息检索系统,识别诸如 *civil suit* 和 *criminal suit* 之类的语义用法之间的差别要比单独列出语义 *to suit* 更为重要。

一些无监督消歧算法的结果在表 7.9 中列出。我们需要考虑这些结果由于不同的初始化引起的变化(见图 7.8 中的第一步)。表中显示了 10 个实验的平均准确率和标准差。对于领域相关性很强的语义,算法的效果非常好并且变化很小。词 *suit* 就是这样的例子。但是,对于带有领域独立性语义的词算法,效果并不好,比如 *train* 的 *to teach* 语义。除了比较差的平均效果之外,领域独立的语义标准差也很高。一般来说,当没有词典资源用来训练和定义语义的时候,无监督算法的效果要比基于词典算法的效果低 5% 至 10%。

表 7.9 无监督消歧算法的一些结果。表中给出了 10 个带有不同 EM 算法初始条件例子的均值 μ 和方差 σ 。数据来自 (Schütze 1998: 110)

词	语义	准确率	
		μ	σ
<i>suit</i>	<i>lawsuit</i>	95	0
	<i>the suit you wear</i>	96	0
<i>motion</i>	<i>physical movement</i>	85	1
	<i>proposal for action</i>	88	13
<i>train</i>	<i>line of railroad cars</i>	79	19
	<i>to teach</i>	55	31

7.5 什么是语义

既然已经考虑过了很多语义消歧的不同方法,现在让我们重新回到究竟什么是语义的问题上来。把语义定义为一个词的不同含义的表达,似乎是很自然的。但是,关于意思的表达方式我们知之甚少,而且我们很难设计一个实验来确定语义是怎样被一个主题表示的。一些研究者根据主题来聚类上下文。针对主题建立了很多索引,每个索引带有一个包含歧义词的句子,并且在主题指导下把这些索引分类到相关的子类中。虽然这个实验给我们提供了很多启示,例如对于研究语义相似性概念是有用的,参见 Miller and Charles (1991),但是我们还是不清楚,在实际的语言理解和生成过程中用这些方法模写词汇和语义,其使用效果如何。确定语言相似性并不是人们在实践中要面对的问题,所以通过不同主题实现聚类的意义不大,参见 (Jorgensen 1990)。

其他很多关于歧义心理学实验的问题依赖于内省和人们通常的主观感觉。我们还不清楚,内省对于了解语义的表达是不是一个有效的方法,因为这类方法在试图说明其他现象的时候都失败了。例如,人们趋向于有理化一些本身无理可言的经济决策,参见 (Kahneman et al. 1982)。

最常用的消歧方法是利用词典中给出的定义,然后根据这些定义标注语料库中的例子。对这种方法的利弊有不同的观点。一些研究者在判断语义时有很高的一致性,和前面讨论的(Gale et al. 1992a)一样。如果有很多的歧义词服从偏斜分布(skewed distribution),那么这样的高一致性是可能的,所谓的偏斜分布是指一个语义被这个词的大多数情况所使用。Sanderson and van Rijsbergen(1998)认为偏斜分布是歧义词的典型分布。

但是像(Gale et al. 1992a)那样随机选择歧义词会带来统计偏差(bias),这就意味着这样的实验结果不能真实反映实际的相互判断一致性。具有语义判断不一致性的很多歧义词是高频词。所以即使在词形(type)基础上语义判断具有比较低的不一致性,在词次(token)基础上也可能有高的不一致性。在最近的一项实验中,Jean Véronis(p.c., 1998)发现在频繁出现的法语词 *correct*, *historique*, *économie* 和 *comprendre* 中,没有一个词的语义判断具有完全的一致性。Véronis 发现主要的原因就是词典定义模糊和语料库中的真实歧义。

我们可以编撰一个语义定义清晰的词典吗? Fillmore and Atkins(1994)从词典编撰的观点讨论了这个问题。一些研究者认为一个词汇的几个语义可以被同时使用或同时激活(co-activated),是词汇意思的内在属性(Kilgariff 1993; Schütze 1997; Kilgariff 1997),正是这个原因导致了相互判断具有很高的不一致性。当然,有些使用多重语义的双关语,如例句(7.9),是特殊情况,自然语言处理系统不能正确判断是可以接受的:

In AI, much of the I is in the beholder. (7.9)

但是,Kilgariff(1993)指出,这样的语义同时被使用的例句在普通语言中很常见。例如在例句(7.10)中,*competition* 的两个语义 the act of competing 和 the competitors 就被激活了。

For better or for worse, this would bring competition to the licensed trade. (7.10)

同时激活(coactivation)的很多例子是系统歧义(systematic polysemy)的情况,同时激活可以作为词汇语义规则,这些规则作用到一类词汇上,并且系统地改变或扩展了这些词汇的意思。读者可参见(Apresjan 1974), (Pustejovsky 1991), (Lakoff 1987), (Ostler and Atkins 1992), (Nunberg and Zaenen 1992)和(Copestake and Briscoe 1995)关于系统歧义的讨论,参见(Buitelaar 1998)近期的计算研究。词汇 *competition* 就是这样的例子,并且有大量的英语词汇在 the act of X 和 the people doing X 二者之间有相同的意思选择,例如 *organization*, *administration* 和 *formation*。

有些系统歧义的类型在实际应用中不能被忽略,比如几乎所有的词都可以被用来作为专有名词,其中有些是很常见的,如 *Brown*, *Bush* 和 *Army*。

对于歧义性程度较高的词汇,较低的相互判断一致性和效果较差的消歧算法仅仅考虑了语义之间区别明显的特征,例如跨语言特征(Resnik and Yarowsky 1998)。系统多义性(systematic polysemy)在很多语言中都是相似的,所以无需区分 *competition* 的两个相关语义(the act of competing 和 the competitors)。这样的策略和自然语言处理中其他领域的一些处理方法是类似的。例如,在句法分析中我们定义了一个较简单的问题:浅层句法分析(shallow parsing),没有试图解决最难的附着歧义问题。

语义消歧中的聚类方法(例如上下文消歧)也使用了相同的策略。通过定义,自动聚类仅仅寻找可以成功消歧的类。这样做相当于把聚类限制到了问题的一个部分。这种规模受限的解决方法非常实用。很多翻译歧义的区别都是不明显的,所以把系统限制在一个粗糙的语义区分上已经足够了。上下文消歧已经成功地应用到了信息检索中(Schütze and Pedersen 1995)。

这类面向应用的语义概念有一个优点,只要含有消歧的应用能被评价,就很容易评价系统语义消歧的效果。例如,机器翻译可以评价系统的翻译准确率,信息检索可以评价系统召回率和准确率。消歧准确率的直接评价和不同算法的比较是非常困难的,但是在将来随着标准评测集的开发,它们将会变得越来越简单。参见 Mooney(1996)中大量机器学习算法的比较评价, Towell and Voorhees(1998)有三个高歧义词(*hard*, *serve* 和 *line*)的消歧效果比较。Senseval 工程中包括了算法的系统评测。

其他一些影响语义概念的因素(隐含的)是消歧中使用的信息类型:同现(可有重复的单词集模型, the bag-of-words model), 相关信息(例如, 主语和宾语等), 其他语法信息(例如词性), 搭配(每个搭配一个语义)和文本(每篇文本一个语义)。例如, 如果仅仅使用同现信息, 那么只有“典型”语义差别可以被识别出来, 这些语义通常相应于不同的领域。Justeson and Katz(1995a)强调在识别很多语义时可有重复的单词集模型是不充分的。Leacock et al.(1998)考虑了主题信息和搭配信息的组合, 使用这两种信息可以得到最佳结果。Choueka and Lusignan(1985)发现, 在很少的形容词上下文的情况下, 人工的语义识别效果出奇得好, 而且在此基础上给出更多的上下文信息对人类的语义消歧效果帮助不大的结论。但是, 这并不意味着更多的上下文信息对计算机消歧是没有用的。Gale et al.(1992b)认为在歧义词周围(每一边)超出 50 个词的上下文中仍然含有对消歧有益的信息, 而且含有对消歧有益信息的上下文范围非常大(大约 1000 个单词)。

不同信息类型最好应用于不同词性的消歧。例如, 动词最好使用它的语义对象(主语和宾语)信息来消歧, 这些信息暗含了很重要的局部信息。很多名词都有区别明显的歧义(如 *suit* 和 *bank*), 所以用来消歧它们的上下文范围应该比较大。

在语义消歧方面依然还有很多内容有待进一步的研究。特别是在“典型”歧义词上评价算法优劣的研究工作变得越来越重要。只有使用更加彻底有效的评价方法, 我们才能够全面地理解本章介绍的消歧算法的优点和缺点。

7.6 深入阅读

关于语义消歧方面的统计和非统计方法研究的一个比较全面的论述可以在(Ide and Véronis 1998)中找到, 也可以参考(Guthrie et al. 1996)。和语义消歧比较相似的工作有句子边界界定问题(sentence boundary identification)(参见 4.2.4 节)。Palmer and Hearst(1997)把这个问题映射为两个句点的消歧问题, 即句点是作为缩写形式中的点号, 还是用于表示句子的结束, 或者两者兼而有之。

本章中我们主要根据不同方法使用的词汇资源的数量和类型来叙述消歧算法。在评述这些方法的时候, 首先涉及到了有监督算法、基于词典的算法和无监督算法, 然后介绍了几种和本章内容结构并不是很相符的方法。

两个重要的有监督消歧学习算法是 K 近邻法(又称基于记忆的学习法(memory-based learning))和线性模型法。(Dagan et al. 1994, 1997b)介绍了最近邻消歧法。他们使用了 K 近邻法的优点来解决数据稀疏问题。参见(Ng and Lee 1996)和(Zavrel and Daelemans 1997)也可了解有关内容。一种可分解的线性模型可以看成是单纯贝叶斯分类器。它不再认为所有特征都是独立的, 而是把特征划分到若干子集中。在这些子集中特征是依赖的, 而不同子集的特征之间是独立的, 这与单纯贝叶斯分类器有所不同, 在单纯贝叶斯分类器中假设所有特征之间都是独

立的。Bruce and Wiebe(1994)使用可分解线性模型进行消歧,得到了很好的结果。

还有一些消歧算法依赖于词汇资源,它们是(Karov and Edelman 1998),(Guthrie et al. 1991)和(Dini et al. 1998)。Karov and Edelman(1998)提出了一种形式体系(formalism)的方法,这种方法综合了语料库和词典资源两方面的优点,并且得到了很好的消歧结果。Guthrie et al.(1991)使用了(Procter 1978)提出的主题领域码(subject field codes),它类似于(Yarowsky 1992)的类义辞典分类方法。Dini et al.(1998)使用基于转换的学习法(transformation-based learning)(参见10.4.1节)来标注带有类义辞典义类歧义的词汇。

介绍聚类方法消歧的文章有(Pereira et al. 1993; Zernik 1991b; Dolan 1994; Pedersen and Bruce 1997; Chen and Chang 1998)。Pereira et al.(1993)使用的聚类歧义词上下文的方法和Schütze(1998)类似,但是聚类的具体方法不同。文中没有在聚类结果上直接使用消歧算法,因为在这样的无监督学习中指定类别和消歧是一样的,消歧应该是聚类的直接扩展结果。14.1.4节有这种方法所用的消歧算法。Chen and Chang(1998)和Dolan(1994)强调了语义的结构表示,把几个子语义(subsense)混合到一个“强语义”(supersense)中。这种子语义聚类方法对于结构化语义(要比词典中提供的粗糙一些)和词典之间的相关语义定义是有益的。

在很多语义消歧方法中都提到了一个很重要的问题,这就是怎样混合不同的证据(McRoy 1992)。(Cottrell 1989; Hearst 1991; Alshawi and Carter 1994; Wilks and Stevenson 1998)中有不同的观点。

虽然在这里我们仅仅涉及到了统计方法,但是在人工智能和计算语言学中研究语义消歧的工作已经有了很长的历史。两篇被经常引用的文章是(Kelly and Stone 1975)(它是一种手工构造的基于规则的消歧方法)和(Hirst 1987)(使用选择约束来消歧)。在前面提到的(Ide and Véronis 1998)中可以找到对于消歧的非统计方法的一般性描述。

7.7 习题

- 7.1 [★]消歧准确性的下界依赖于可以使用的信息有多少。试描述一种情况,在这种情况下,消歧的下界要比直接指定歧义词的语义为最频繁出现语义的做法的消歧下界还要低。提示:在计算下界时需要什么样的知识?
- 7.2 [★★]有监督的语义消歧算法非常容易设计和训练。实现上面讨论的算法之一或者实现一个你自己设计的算法。实现的算法消歧效果怎样? 训练数据可以从Linguistic Data Consortium(DSO语料库)和词网(WordNet)得到,在网站上有些数据资源的链接。
- 7.3 [★★]使用伪词创建一个人造的训练和测试集合,并依此评价一个有监督的消歧算法。
- 7.4 [★★]从本书配套网站上下载一个 Roget 类义辞典(参见本书配套网页),实现和评价一个基于类义辞典的消歧算法。
- 7.5 [★★]我们介绍的两种有监督消歧方法有不同的维度(dimension),这是因为使用的特征数目不同(一个和很多),同时它们使用的数学方法也不同(信息理论和贝叶斯分类)。你可以设计出一个使用一个特征的贝叶斯分类器和一个使用很多特征的信息论方法吗?
- 7.6 [★★]根据我们对于紧密相关语义和同时激活语义的论述,讨论一下伪词模型可以在多大程度上解决这样的歧义。

- 7.7 [★★] Lesk 算法统计了语义定义和上下文中同现词汇的数量。这样做不是最好的解决方法,因为“非描述性词”和停止词(像 *try* 和 *especially*)会导致错误的分类。试着调整 Lesk 算法,根据词汇的区分度来加权词汇。
- 7.8 [★] 信息理论消歧方法和 Yarowsky (1995) 算法都只使用了一个特征。讨论这两种方法的不同和相似性。
- 7.9 [★] 讨论约束“每篇文章一个语义”对不同的消歧类型(用法类型、同音异义词等)的合理性。构造这个约束有效和无效的一些例子。
- 7.10 [★★] 在一个语料库上评价“每篇文章一个语义”的约束。找一些一个歧义词多次使用的文章或段落,看看这些歧义词在一篇文章和段落中有不同语义的情况有多少?
- 7.11 [★] 在无监督消歧算法部分,我们给出了确定一个歧义词语义数量的标准。你认为还有其他标准吗? 假设(a)可以使用词典(但是处理的歧义词不在这个词典中);(b)可以使用类义辞典(处理的歧义词也不在类义辞典中)。
- 7.12 [★] 对于你熟悉的两种语言,找到三个这样的例子:在第一种语言中的歧义词的一个语义可以被翻译成第二种语言中不同的词;另外找三个相反的例子:一个词歧义的至少两个语义可以被翻译成另一种语言中相同的词。
- 7.13 [★] 有必要在一个单独的测试集上评价无监督学习算法吗? 或者根据无监督学习的特性,是否有必要区别训练集和测试集? 提示:在单独测试集上测试是有必要的,具体原因参见 (Schütze 1998:108)。
- 7.14 [★] 在本章开始部分讨论的 *title* 的几种语义和系统歧义 (systematic polysemy) 是有关的。找到其他一些带有相同系统歧义的词。
- 7.15 [★★] 请读者选择一种消歧算法,使用它来确定句子边界。

第8章 词汇获取

在第5章中我们介绍了搭配、短语和词汇组合获取的问题。在自然语言处理中它们或者具有专门的含义,或者具有某种很特殊的用途。在本章中,我们将进一步扩大讨论的范围,考虑更复杂的词汇的句法和语义特征的获取。词汇获取的一般性目的是:通过考察大型文本语料库中词汇的出现模式,设计一种算法和统计技术来填补现有电子词典的不足。在词的搭配问题之外我们还面临着很多其他词汇获取的问题:选择倾向性(例如,动词 *eat* 通常要用食物类的词作为直接宾语),子范畴框架(例如, *contribute* 的受体通常被表示成 *to* 的前置词短语)和语义范畴(词典中没有的新词的语义范畴是什么?)。前面我们已经讨论了如何让计算机直接从在线文本中自动学习词汇信息,虽然这种方法在一定程度上讲相当有效,但它违背了经典的乔姆斯基学派的观点,即由于感官刺激贫乏,人类天生具备语言能力。

在自然语言处理中我们感兴趣的大部分词的特性并没有被收录到电子词典中,这是由自然语言的多产性所导致的。新词和旧词的新用法总是层出不穷,即使我们编辑一个词典,使其可以完全覆盖今天的语言,但是大约一个月以后就会发现它必然会变得不全面,从而不能完全描述当前的语言特性。正是由于这样的原因,词汇获取在统计自然语言处理中才显得如此重要。

在这里我们对使用 *lexical* 和 *lexicon* 所表达的意思进行简短的讨论。Trask (1993: 159) 定义 *lexicon* 为:

That part of the grammar of a language which includes the *lexical entries* for all the words and /or morphemes in the language and which may also include various other information, depending on the particular theory of grammar.

一种语言语法的一部分,包括了反映这种语言中所有词汇和(或)词素的词条,而且也可以包含各种其他信息,这些信息依赖于特殊的语言理论。定义的第一部分(“反映所有词语的词条”)指出,我们可以把 *lexicon* 看成是电子词典的一种扩展形式,它也是机器可读的。麻烦的是,传统词典是针对人类的阅读需求而不是计算机的需要编撰的。数量信息在传统词典中完全没有,因为这些信息对于人类读者根本没用。所以统计自然语言处理中词汇获取的一个重要任务就是在传统词典中增加数量信息。

定义的第二部分(“各种其他信息,依赖于特殊的语言理论”)反映了在词典信息和非词典信息之间没有明显的界限。一个普通的句法规则,比如 $S \rightarrow NP VP$ 就是一个明确的非词典信息,但是前置词短语的附着歧义问题呢? 在某种意义上讲,这是一个句法问题,但是可以通过考察动词和名词(有可能作为前置词短语)的词汇特征得到解决,就像下面的例句:

- a. The children ate the cake with their hands.
 - b. The children ate the cake with blue icing.
- (8.1)

我们从语料中可以学习 *eat* 的一般属性,即它是指可以用手来做的某一类事情,并且 *cake* 也可以和 *icing* 一起作为宾语。在得到了这些词语(*ate and hands* 和 *cake and icing*)之间的依赖关系

之后,我们就能正确地解决例句(8.1)中的附着歧义问题:*with their hands* 附着于 *ate*, *with blue icing* 附着于 *cake*。

在某种意义上讲,几乎所有统计自然语言处理(包括参数估计)都和词汇属性有关,因此很多统计自然语言处理工作都需要进行词汇获取。事实上,有一些语言学理论就宣称所有语言知识都是关于词汇的知识,如依存语法(Dependency Grammar)(Mel'čuk 1988)、直接语法(Categorical Grammar)(Wood 1993)、邻接树语法(Tree Adjoining Grammar)(Schabes et al. 1988; Joshi 1993)和基本词汇(Radical Lexicalism)(Karttunen 1986)。并且,所有的关于语言的知识都可以在词典(lexicon)中体现出来,因此无需把语法作为一个独立的实体。通常,在单个词的级别上可以简单概念化的属性都能包含在词汇获取(lexical acquisition)中。在前面的一些章节中,我们已经讨论了搭配获取和简单的语义消歧,因为这些问题彼此独立并且可以在自然语言中分别处理。然而,从某种意义上讲,它们只是本章讨论词汇获取问题的一些特例。

本章中主要讨论的四部分内容包括:动词子范畴(动词表现的句法含义)、附着歧义(参见例句(8.1))、选择倾向(动词语义对象的语义特征,比如吃的东西通常是食物这样的事实)以及词汇之间的语义相似性。我们首先介绍一些评价方法,它们经常被用来评价词汇获取方法和各种其他的统计自然语言处理系统的性能。在本章结尾我们会深入讨论词汇获取在统计自然语言处理中的重要意义,并给出一些相关的深入阅读资料。

8.1 评价方法

当前,更严格的系统评价标准的使用是自然语言处理的一项重要发展,这个评价标准认为系统最终是否成功,取决于在具体应用上系统的效果是否得到提高,比如拼写纠错、工作广告摘要或其他应用。但是,在开发系统的时候,为了方便起见,我们经常使用人工的效果参数,比如混乱度(perplexity),来评价系统的各个组成部分,我们希望各个部分的效果的提高可以反映基于某一应用的整个系统效果的提高。

在信息检索中的评价经常使用精确率(precision)和召回率(recall)的概念,而且这样的评价方法已经应用到了评价统计自然语言处理模型的工作中,取得了很好的效果。在本章中我们将讨论很多这样的系统。一般来说,在多数问题中,我们都定义了一个目标集合(例如,目标相关文档集或包含一个词的某一个确定语义的句子集),这个集合包含在一个更大的集合中。我们的系统要在一个选择集合(系统认为相关的一些文档,或包含了一个词的某一个语义的一些句子)上做出决策。在图 8.1 中显示了这样的情况。选择和目标集合可以被认为是随机变量,并且这两个变量的联合分布可以表示成一个 2×2 的联立矩阵(contingency matrix):

系统	真实的	
	target	$\neg$ target
selected	<i>tp</i>	<i>fp</i>
$\neg$ selected	<i>fn</i>	<i>tn</i>

(8.2)

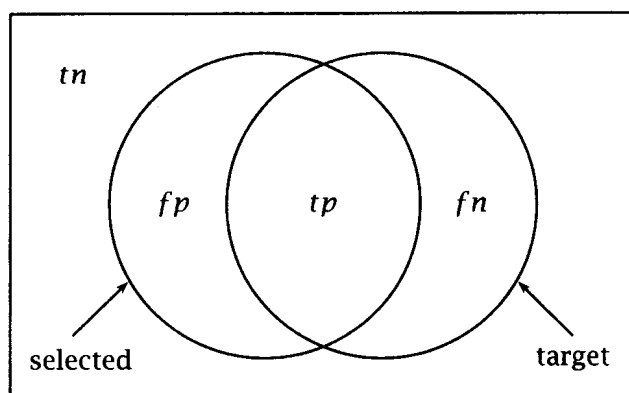


图 8.1 精确率和召回率度量图。图中的区域是按照目标集和选择集表示的。精确率大小为 $tp/|selected|$ ，表示目标项（或正确项）在选择集（或找到项）中的百分比。召回率大小为 $tp/|target|$ ，它表示目标项在目标集中的百分比。同时， $|selected| = tp + fp$ ， $|target| = tp + fn$

矩阵(8.2)中每一项数字表示了图 8.1 的空间中每个区域项目出现的频率或次数。由 tp （真正确, true positive）和 tn （真错误, true negative）表示系统正确做出判断的情况。由 fp 表示的错误选择情况称为假正确（false positive），假接受（false acceptance）或错误类型 II（type II error）。由 fn 表示的选择失败的情况称为假错误（false negative），假拒绝（false rejection）或错误类型 I（type I error）。

精确率（precision）被定义为系统选择正确项在全部选择项目中所占的比例：

$$\text{precision} = \frac{tp}{tp + fp} \quad (8.3)$$

召回率（recall）被定义为系统选择的目标项在全部目标项中所占的比例：

$$\text{recall} = \frac{tp}{tp + fn} \quad (8.4)$$

在信息抽取应用中，我们通常会协调精确率和召回率（可以选择文档集中的所有文档，得到 100% 的召回率，但是精确率非常低）。这种协调可以绘制到精确率 - 召回率曲线中，就像我们在 15.1.2 节中所做的那样。有时这种协调在自然语言处理应用中没有任何意义。但是如果存在这样的情况，例如对一些统计项比其他一些更有把握时（例如 8.2 节中的子范畴框架学习），我们可以继续协调精确率和召回率。

由于这个原因，我们可以方便地把精确率和召回率统一到一个全面的度量尺度中。一种方法是使用 F 测量（ F measure），它是 van Rijsbergen(1979: 174)发明的 E 测量（ E measure）的变体，在这里 $F = 1 - E$ 。 F 测量的定义是：

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} \quad (8.5)$$

其中 P 表示精确率， R 表示召回率， α 表示确定精确率和召回率权重的因子。 $\alpha = 0.5$ 对应着选择相同权重的 P 和 R 。对应 α 的这个值， F 测量可以简化为 $2PR/(P + R)$ 。

一个很好的问题是：“在矩阵(8.2)中， $tp + tn$ 是事件正确的数目， $fp + fn$ 是事件错误的数

目。为什么我们不报告事件正确的百分比或事件错误的百分比?”的确我们可以这样做的,而且这种度量方法一般被称为准确率(accuracy)和错误率(error)。但是像这样的度量方法事实上并不实用,因为在大部分问题框架中, tn 的值(非目标,非选择事件的数目)都非常大,从而会使其他数据显得微不足道。针对这种情况,我们一般都使用精确率和召回率作为度量的方法,它具备三个优点:

- 准确率数值对于比较小但我们感兴趣的数字 tp , fp 和 fn 不是特别敏感,而精确率和召回率对于这些数字非常敏感。你可以简单地什么也不选,但是会得到一个非常高的准确率。
- 在其他条件相等的情况下, F 测量和真正确(true positive)的数量成正比,而准确率只对错误的数量很敏感。 F 测量这种倾向和我们的直观感觉是一致的:我们对发现事件感兴趣,甚至返回一些垃圾数据也在所不惜。
- 我们可以看出,遗漏的目标事件和所谓垃圾事件对于系统性能的影响其实并不相同,而利用精确率和召回率可以衡量出其中的差异。

表 8.1 中给出了一些事例,阐明了准确率和 F 测量($\alpha = 0.5$)评价结果的不同。

表 8.1 F 测量和准确率是不同的客观方程。表中显示的精确率、召回率、 F 测量($\alpha = 0.5$)和准确率是对一个 100 000 项数据集进行选择的结果,在这个数据集中有 150 项数据是目标数据。表的上半部分(a)显示了随着 F 测量的增加,准确率在下降。下半部分(b)有同样大小的准确率,但是 F 测量的值仍在增加。 F 测量的倾向是最大化真正确(true positive),而准确率仅仅对分类错误的数字敏感

	tp	fp	fn	tn	Prec	Rec	F	Acc
(a)	25	0	125	99 850	1.000	0.167	0.286	0.9988
	50	100	100	99 750	0.333	0.333	0.333	0.9980
	75	150	75	99 700	0.333	0.500	0.400	0.9978
	125	225	25	99 625	0.357	0.833	0.500	0.9975
	150	275	0	99 575	0.353	1.000	0.522	0.9973
(b)	50	0	100	99 850	1.000	0.333	0.500	0.9990
	75	25	75	99 825	0.750	0.500	0.600	0.9990
	100	50	50	99 800	0.667	0.667	0.667	0.9990
	150	100	0	99 750	0.600	1.000	0.750	0.9990

一个不太常用的评价方法是漏识率(fallout),它表示了被错误选择的非目标项在非目标集中所占的百分比,也就是本应该是错误的现在却被系统判断为正确的:

$$\text{fallout} = \frac{fp}{fp + tn} \quad (8.6)$$

当某个系统只产生很少的假正确(false positive)情况时,漏识率可以用来评价这个系统构建的困难程度。如果非目标项的数目非常大,那么由于 fp 较大而产生较低的精确率就可能无法避免,因为在非目标项庞大的对象总体下,一些项被错误分类总是不可避免的。

在一些工程领域中,recall-fallout 之间的协调比 precision-recall 之间的协调更加常用。我们可以使用一个所谓的 ROC 曲线(对于接受器操作特点)来显示不同级别的 fallout 度量值怎样影响召回率或敏感性(sensitivity)(真正确在所有目标事件中占有的比例),就像防盗警报器有

一个调节灵敏度的铃舌一样。ROC 曲线会告诉我们,对于一个确定的假正确率,预期的真正正确率是多少。例如,在 100 个晚上有一次警报器报警,但是没有盗贼,通过这个假正确率,我们也许可以得到一个真正正确的期望比率:95%(也意味着盗贼有 5% 的机会不被发现)。

- ▼ 在这个概率分析中使用的评价方法将在 12.1.8 节中讨论,而且对 IR 的评价方法也会在 15.1.2 节中进一步讨论。

8.2 动词子范畴

我们在 3.2.2 节中已经指出,动词可以被划分为不同的句法范畴,也就是说,动词可以用不同的句法形式来表达自己的语义对象。我们把句法范畴集合称为子范畴框架(subcategorization frame)。表 8.2 中给出了一些子范畴框架的例子。由于英语动词总是会有一个主语,所以我们有时在句法框架中省略主语。

表 8.2 带有示例动词和例句的子范畴框架,改编自(Brent 1993: 247)

框架	功能	动词	例句
NP NP	subject, object	greet	<u>She</u> greeted <u>me</u> .
NP S	subject, clause	hope	<u>She</u> hopes <u>he will attend</u> .
NP INF	subject, infinitive	hope	<u>She</u> hopes <u>to attend</u> .
NP NP S	subject, object, clause	tell	<u>She</u> told <u>me</u> <u>he will attend</u> .
NP NP INF	subject, object, infinitive	tell	<u>She</u> told <u>him</u> <u>to attend</u> .
NP NP NP	subject, (direct) object, indirect object	give	<u>She</u> gave <u>him</u> <u>the book</u> .

之所以把这样的现象叫做子范畴,是因为我们把带有特殊语义论元集的动词看成是一个范畴。进一步讲,这些范畴又可以分为几个子范畴,子范畴使用不同的句法结构来表示它们的语义对象。例如,带有语义对象 theme(主题)和 recipient(接受器)的一类动词可以有这样的子范畴,这个子范畴使用宾语和前置词短语来表示它们的语义论元(例如, *donate* 在例句 *He donated a large sum of money to the church* 中),并且还可以有其他的子范畴,这个子范畴允许有一个双宾语结构(例如, *give* 在例句 *He gave the church a large sum of money* 中)。

了解动词的子范畴框架对句法分析非常有用。通过对比例句(8.7)中的句子,读者就可以知道其中的原因。

- a. She told the man where Peter grew up.
b. She found the place where Peter grew up. (8.7)

如果知道了 *tell* 有一个子范畴框架:NP NP S(主语,宾语,从句),并且知道 *find* 没有这样的子范畴框架,但是 *find* 有一个子范畴框架:NP NP(主语,宾语),我们就能对 *where* 从句的所属问题得出正确的结论。在第一个句子中,*where* 从句属于 *told*,如例句(8.8a)所示,第二个句子中的 *where* 从句属于 *place*,如例句(8.8b)所示。

- a. She told [the man] [where Peter grew up].
b. She found [the place [where Peter grew up]]. (8.8)

遗憾的是,大部分词典都没有包含子范畴框架的信息。即使在很少几个包含子范畴框架信息的词典中,如(Hornby 1974),大部分动词的信息也是不完全的。根据一项统计,有超过 50% 的

句法分析错误是由于缺少子范畴框架引起的^①。最全的英语子范畴框架信息资源可能就是(Levin 1993)了,但是它也不能完全覆盖所有的子范畴框架,并且也不包含数量信息,比如一个动词不同的子范畴框架出现的相对频率。而且,语言生成也需要从语料库中获取一些子范畴的信息。

Brent(1993)提出了一个简单有效的学习子范畴框架的算法,实现该算法的系统称为 Lerner。假设我们基于一个语料库来判断动词 v 是否有框架 f ,为了解决这个问题,Lerner 系统将分两步来做出决策:

- **暗示(cue)**:定义了包含词语和句法范畴的正则模式,该正则模式表明某个特定框架的出现有比较高的确定性(certainty)。确定性可以形式化为错误概率。对于某个特定的暗示 c ,我们定义了一个错误概率 ϵ_c ,表示如果在暗示 c 的基础上为动词 v 指定框架 f ,那么错误的可能性有多大。
- **假设检验(hypothesis testing)**:它的基本思想为:我们最初假设框架不适合这个动词,并把它作为零假设 H_0 。如果暗示 c 指示 H_0 错误的概率很高,那么我们拒绝这个假设。

暗示:暗示实际上就是正则模式,Brent(1993: 247)使用这些正则模式来发现子范畴框架“NP NP”(及物动词):

$$\begin{aligned} &\text{Cue for frame "NP NP":} \\ &(\text{OBJ} \mid \text{SUBJ_OBJ} \mid \text{CAP}) (\text{PUNC} \mid \text{CC}) \end{aligned} \quad (8.9)$$

在这个暗示中,OBJ 表示只能作为宾格(或宾语)的人称代词,如 *me* 和 *him*;SUBJ _ OBJ 表示既可以作为主语又可以作为宾语的人称代词,如 *you* 和 *it*,CAP 表示任何以大写字母开头的单词;PUNC 是标点符号;CC 是主从连接词,如 *if*, *before* 或 *as*。

我们之所以选择这个模式,是因为它只可能出现在一个动词具有“NP NP”框架的时候。假设我们有类似例句(8.10)的句子,这个句子对应的实例化模式就是“CAP PUNC”:

$$[\dots] \text{greet-V Peter-CAP, -PUNC} [\dots] \quad (8.10)$$

当然,我们也可以假设有例句(8.11)这样的句子,使之匹配模式(8.9),但是动词却并不具有框架“NP NP”的特征。在例句(8.11)中对应的模式是 *came-V Thursday-CAP, -PUNC*。这种情况是非常少见的,因为当一个动词接一个大写字母开头的词,并且在接一个标点符号时,它几乎总是只有一个宾语而且不再需要其他的句法对象(当然还需要主语)。所以当暗示(8.9)出现的时候,动词具有框架“NP NP”的错误概率很低。

$$\text{I came Thursday, before the storm started.} \quad (8.11)$$

在暗示的真实性和广泛性之间,我们需要做一个折中。模式“OBJ CC”可能就比“CAP PUNC”更不容易导致错误。但是如果我们把模式(8.9)减少到只存在一个可信的实例,则将不得不通过大量动词的例子来寻找这个暗示出现的实例,这也将会导致算法测试只能针对那些出现非常频繁的动词。这个问题在下文中还将会讨论到。

假设检验:当框架的暗示定义好以后,我们就可以对语料库进行分析,并且对于任意动词

① 参见 John Carroll 在 workshop “Practical Acquisition of Large-Scale Lexical Information”(at CSLI, Stanford, on April 23, 1998) 中给出的论文“Automatic acquisition of subcategorization frames and selectional preferences from corpora”。

和框架的组合,可以统计出这个框架的暗示和这个动词同现的次数。假设动词 v^i 在语料库中总共出现了 n 次,并且有 $m \leq n$ 次是和框架 f^j 的暗示同现的。那么我们就可以使用下面给出的错误概率拒绝零假设 $H_0(v^i$ 不具有框架 $f^j)$:

$$p_E = P(v^i(f^j) = 0 | C(v^i, c^j) \geq m) = \sum_{r=m}^n \binom{n}{r} \epsilon_j^r (1 - \epsilon_j)^{n-r} \quad (8.12)$$

在公式中, $v^i(f^j) = 0$ 表示动词 v^i 不具有框架 f^j , $C(v^i, c^j)$ 是 v^i 和 c^j 的同现次数, ϵ_j 是 c^j 的错误概率,这个概率是指和 c^j 同现但不具有框架 f^j 的动词出现的概率。

在第5章中我们已经讨论了零假设检验的基本思想: p_E 是零假设成立时观测数据的概率。如果 p_E 比较小,那么我们拒绝零假设 H_0 , 因为一个不太可能的事件出现表明了假设 H_0 是错误的。在这个推理过程中我们的错误概率为 p_E 。

在式(8.12)中,我们假设了一个二项分布(参见2.1.9节)。动词的每一次出现都是一个独立掷硬币的随机过程。对于这个过程,暗示错误的概率为 ϵ_j (也就是说,暗示出现,但是动词不使用暗示的框架),暗示正确的概率值大小为 $1 - \epsilon_j$ (暗示和指示框架一起出现,或者两者都没有出现)^①。因此可以断定,如果我们发现了 m 次或者更多次的框架暗示,拒绝 H_0 的错误概率就是 p_E 。如果对于一个合适的置信度 α 有 $p_E < \alpha$, 那么我们就可以拒绝零假设,例如, $\alpha = 0.02$ 。如果 $p_E \geq \alpha$, 我们就认为动词 v^i 不具有框架 f^j 。

实验评价表明,以精确率作为度量, Lerner 系统效果很好。对于大部分子范畴框架,带有特殊框架的动词有接近 100% 的精确率 (Brent 1993: 255)。然而, Lerner 系统的召回率并不是很理想。对于 Brent (1993) 提出的 6 个框架, 它的召回率在 47% 到 100% 之间。但是, 如果我们使用动词词形 (verb type) 代替动词词次 (verb token), 那么该系统的性能可能会稍微差一些, 因为采用这样的采样方法后低频动词可能只占很小的比例^②。然而, 我们知道低频动词被收入现有词典的可能性很小, 所以相对于高频动词来说, 正确得到它们的框架显得更为重要。

Manning (1993) 阐述了低召回率的问题, 使用了标注器, 并且在标注器的输出上使用暗示决策。暗示决策就是对应模式 (8.9) 的正则表达式。这样做似乎会导致出现一些问题, 因为我们现在有了两个错误倾向 (error-prone) 系统, 标注器和暗示决策, 它们混合在一起并产生了一个更有错误倾向的系统。然而, 在假设检验中这是没有问题的。这主要是因为, 暗示作为一个子范畴指示器的可靠性有多大是无关紧要的。如果一个不可靠的指示器出现得足够频繁, 作为一个合适的假设检验, 这个指示器还可以帮助我们确定动词的子范畴框架。例如, 一个错误率 $\epsilon_j = 0.25$ 的暗示出现了 11 次 (在 80 次中), 那么我们仍然可以拒绝零假设 (动词 v^i 不具有 c^j), 因为 $p_E \approx 0.011 < 0.02$, 虽然 c^j 的可靠性很低。

低可靠性暗示和额外的基于标注器输出的暗示可能会极大地增加可使用暗示的数量。这样做的结果是出现了更多的动词具有符合给定框架的暗示。而且, 更重要的是, 很多子范畴框架都具有可靠性不高的暗示, 例如前置词子范畴, 比如 *on* 在 *he relies on relatives* 中或者 *with* 在 *she compared the results with earlier findings* 中。因为大部分出现在动词后面的前置词不是子范畴

① 我们在这里省略了 Lerner 系统的第 3 部分: 确定每一个框架的 ϵ_j 的方法。有兴趣的读者可以参考 (Brent 1993)。

② Brown 语料库中动词的每次出现和其出现在样本中的概率是相同的, 所以这个样本偏向中高频动词。

的暗示,所以这些暗示作为动词前置词子范畴的暗示就不太可靠。使用 Manning 的方法可以学习更多的子范畴框架,甚至可将其用于只有低可靠性的暗示的框架。

表 8.3 中显示了 Manning 结果的一个采样。我们可以看到其精确率是很高的:只出现 3 个错误。其中的两个错误是前置词短语简称 PP: *to bridge between* 和 *to retire in*。一个前置词短语是语义论元(被子范畴化)还是一个附着短语,实际上是很难严格区分的。我们可以认为,在 *John retires in Malibu* 这样的句子中, *retire* 被子范畴化为 PP *in Malibu*, 因为动词和 PP 补语之间的关系比单纯的状态修饰关系更紧密一些。例如,我们可以推测出 John 结束了冲浪生活很长一段时间了。但是 OALD(Oxford Advanced Learner's Dictionary)(Hornby 1974)并没有把“NP in-PP”列为子范畴框架。

表 8.3 Manning 系统学习得到的一些子范畴框架。对于每一个动词,表中显示了学习出的正确的和不正确的子范畴框架的数目,还有 OALD 列出的框架数目。改编自 (Manning 1993)

动词	正确	不正确	OALD
<i>bridge</i>	1	1	1
<i>burden</i>	2		2
<i>depict</i>	2		3
<i>emanate</i>	1		1
<i>leak</i>	1		5
<i>occupy</i>	1		3
<i>remark</i>	1	1	4
<i>retire</i>	2	1	5
<i>shed</i>	1		2
<i>troop</i>	0		3

表中的第 3 个错误是把不及物框架错误地指定给了 *remark*。这可能是由类似例句(8.13)这样的句子造成的,在这个句子中 *remark* 看起来似乎没有任何语义论元(除了主语)。

“And here we are 10 years later with the same problems,” Mr. Smith remarked. (8.13)

在表 8.3 中给出的数据的召回率相对较低。此处召回率是指正确识别的子范畴框架占 OALD 中列出的子范畴框架数目的百分比。高精确率和低召回率是我们使用假设检验的必然结果。我们仅仅寻找已经被很好地证明了的子范畴框架。相反,这也意味着我们不会理会那些出现极少的子范畴框架。例如,在 *he leaked the news* 中 *leak* 的及物动词用法,这样的框架就不会被我们发现,因为它在语料库中出现的数目不够多。

表 8.3 仅仅是一个实例抽样。在 Manning 的实验中总共有 40 个动词,精确率是 90%,召回率是 43%。加入动词子范畴框架的先验知识可以用于改善实验的结果。虽然无需任何词典知识的帮助就可以从生语料中学习子范畴框架是很吸引人的,但是如果我们使用了词典中的先验知识,结果无疑将会大大改善。例如,保证词典中列出的子范畴框架具备比较高的优先权,就是指定先验知识的一个最简单的方法。

下面我们举例说明先验知识是怎样提高准确率的。假设我们分析一个特殊的句法模型(比如 V NP S),并且找到了两个可能的子框架模型 f^1 (subject, object)和 f^2 (subject, object, clause), f^2 的概率稍微高于 f^1 。这实际上就是例句(8.8)。句法分析器将会为一个动词选择 f^1 ,对这个

动词来说两个框架具有相同的优先权,如果一个动词借助于先验知识否定了其中的一个框架(比如 f^1),我们很自然地就会为这个动词选择框架 f^2 。例如,如果知道 *email* 是一个通信类动词(类似 *tell*),我们就可能不太倾向于不带从句的框架,因此句法分析器就会正确地选择框架 f^2 (subject, object, clause)给例句 *I emailed my boss where I had put the file with the slide presentation* 中的 *email*。这样,一个基于不完全子范畴化词典的系统将会更好地利用语料库,它比直接在语料库上学习更高效,因此结果也会更好。

8.3 附着歧义

附着歧义问题是自然语言分析过程中普遍存在的一个问题。当我们分析一个句子的句法结构时,经常会遇到这样的问题,即有些短语可以附着于句法树上的两个或者多个节点,因此我们不得不确定这些短语的附着关系。介词短语附着就是一个被广泛研究的附着歧义问题。下面我们给出一个介词短语附着歧义的例句(8.14),它是第3章的例句(3.65):

The children ate the cake with a spoon. (8.14)

介词短语 *with a spoon* 在句子中附着的位置不同,其含义也不同:孩子使用 *spoon* 吃蛋糕(介词短语附属于 *ate*),或者是孩子吃的是一个插着 *spoon* 的蛋糕(介词短语附属于 *cake*)。后一种理解是这个介词短语的很特殊的一种用法,但是对于像 *with frosting* 这样的介词短语,这种用法就很正常。图 3.2 中的两个不同的句法树对应了这两种不同的附着关系。如果句子中介词短语出现在作为宾语的名词短语后面,一般都会产生这种类型的句法结构歧义。正是这种歧义结构导致例句(1.12)中的句子存在不止一棵句法分析树,因为在这个句子中有大量的介词短语存在(还有关联子句),它们产生了非常多的附着歧义问题。在本节中,我们将会介绍一种基于词典信息的解决介词短语附着歧义的方法,由 *Hindle and Rooth* (1993)提出。

那么应该怎样解决这样的歧义问题呢?我们可以设想把这个例子放到上下文中去考虑,介词短语 *with a spoon* 可以起到区分不同 *cake* 的作用,从这一点我们可以看出该介词短语作为“吃”的工具更自然一些,因此更有可能附着于动词。对于很多以较为自然的方式组织的句子,这种做法是正确的:

- a. Moscow sent more than 100 000 soldiers into Afghanistan ...
b. Sydney Water breached an agreement with NSW Health ... (8.15)

在这些例句中,只有一种附着形式是合理的。在例句(8.15a)中,介词短语 *into Afghanistan* 应该附着到以 *send* 为中心词的动词短语上,而在例句(8.15b)中,介词短语 *with NSW Health* 应该附着到以 *agreement* 为头词的名词短语上。一般来说,词汇的选择的偏向性可以用来解决这类消歧问题。当然,退一步讲,在大部分情况下,利用简单的词汇统计就能确定哪一种附着关系是正确的。这些简单的统计包括基于动词和前置词的同现计数和基于名词和前置词的同现计数。在语料库中,我们可以找到很多 *into* 和 *send* 同现的例子,但是 *into* 和 *soldier* 同现的次数相对较少,所以我们有充分的理由相信以 *into* 开头的介词短语附着于 *send* 而不是 *soldier*。

基于这些信息的一个简单模型使用了似然比(likelihood ratio) λ (和 5.3.4 节中的似然比相似):

$$\lambda(v, n, p) = \log \frac{P(p|v)}{P(p|n)} \quad (8.16)$$

其中 $P(p|v)$ 表示带有 p 的介词短语出现在动词 v 后面的概率, $P(p|n)$ 表示带有 p 的介词短语出现在名词 n 后面的概率。 $\lambda(v, n, p) > 0$ 说明介词短语附着于动词, $\lambda(v, n, p) < 0$ 说明介词短语附着于名词。

这个模型的缺点是它简单地认为其他因素对于确定附着关系都是相等的。事实上,在句法分析树中存在着一个附着“偏低”的倾向问题。对于介词短语的附着问题,在句法树中偏低的节点是名词短语(NP)节点。例如,在图 3.2(b)中的句法树上,介词短语 *with a spoon* 附着于偏低的名词短语节点,在图 3.2(a)中附着的是偏高的动词短语(VP)节点。对于这个例子,我们可以这样解释偏低附着倾向。当我们处理介词短语时,名词短语刚刚被处理完,还没有被忘记,所以我们很容易会把这个介词短语倾向于附着到名词短语上。

下面给出一个例子(取自纽约时报),表明了我们考虑这种附着倾向性的重要性:

Chrysler confirmed that it would end its troubled venture with Maserati. (8.17)

上面例句中的介词 *with* 出现在 *end* (例如 *the show ended with a song*) 和 *venture* (例如 *the venture with Maserati*) 后面的情况都很多。我们从纽约时报语料库中得到的数据列在表 8.4 中^①,当我们使用式(8.16)处理完这些数据后,判定的结果是该介词短语附着于动词:

$$P(p|v) = \frac{607}{5156} \approx 0.118 > 0.107 \approx \frac{155}{1442} = P(p|n)$$

表 8.4 介词短语附着消歧出错的一个例子

w	$C(w)$	$C(w, with)$
<i>end</i>	5156	607
<i>venture</i>	1442	155

但是,在本例中这是一个错误的判断结果。模型之所以出错是因为式(8.16)忽略了所谓低附着偏向性的情况,也就是说,当介词短语既可以附着于动词也可以附着于名词时,它偏向于句法树中位置较低的节点。我们将会在下面给出一个概率模型,其中包括了这种偏向性的正式定义。

8.3.1 Hindle and Rooth(1993)的概率模型

在介绍 Hindle and Rooth(1993)的概率模型之前,我们首先定义事件空间的概念。我们所感兴趣的句子都可能存在介词短语附着歧义问题,所以事件空间是由所有符合下列条件的子句组成的:要求子句中包含一个及物动词(有名词短语作为宾语的动词),有一个在动词之后的名词短语(宾语名词短语),和一个在名词短语之后的介词短语^②。我们的目的就是解决这种情况下的介词短语附着歧义问题。

为了减少模型的混乱度(complexity),每次只考虑一个介词(也就是说,忽略了以不同介词开头的介词短语之间的相互关系,参见习题 8.8)。而且,如果在句子中依次出现两个含有相同介词的介词短语,那么我们也只针对第一个介词构建模型(参见习题 8.9)。

① 我们使用的是第 5 章的文本子集。

② 在这里,我们的术语概念比较宽泛,因为当介词短语(PP)附着于一个名词时,它其实是这个名词短语(NP)的一部分,所以严格说来,它不能跟在名词短语后面。因此,我们在这里说的名词短语(NP)是不带补充和修饰成分的基本名词短语(base NP)。

出于简化概率模型的目的,我们并不直接以确定的形式来判断介词是否附着于某个动词或名词,而是给出在一般情况下一个介词附着于一个动词或名词的可能性估计,即以概率的形式来描述附着的可能性。考虑下面两个问题,借助于随机变量指示器 VA_p 和 NA_p 的集合,我们给出它们的形式化定义:

VA_p : 是否存在一个在动词 v 后面以 p 开头的介词短语附着于 v ($VA_p = 1$) 或者不附着于 v ($VA_p = 0$)?

NA_p : 是否存在一个在名词 n 后面以 p 开头的介词短语附着于 n ($NA_p = 1$) 或者不附着于 n ($NA_p = 0$)?

需要注意的是,我们在这里指的是介词 p 的任何一种可能情况,而不是它的一个特殊实例,所以对于一个 p 值, VA_p 和 NA_p 可能都等于 1。例如,在下面的句子中 $p = on$ 就是这样的情况:

He put the book [on World War II] [on the table]. (8.18)

对于包含序列“ $v \cdots n \cdots PP$ ”的句子,我们希望计算以介词 p 开头的介词短语(PP)附着于动词 v 和名词 n 的概率,以 v, n 作为条件:

$$P(VA_p, NA_p | v, n) = P(VA_p | v, n) P(NA_p | v, n) \quad (8.19)$$

$$= P(VA_p | v) P(NA_p | n) \quad (8.20)$$

在式(8.19)中我们假设两个附着关系是条件独立的,也就是说,介词短语是否修饰动词 v 和是否修饰名词 n 是独立的。在式(8.20)中,我们假设动词是否被介词短语修饰与名词无关,而且名词是否被介词短语修饰也与动词无关。

我们把一个介词附着于一个动词和一个名词的附着关系(例如 VA_p 和 NA_p)看成是独立的,这似乎是不合情理的,因为如前所述,介词短语在名词和动词附着关系上是二值选择的(非此即彼),所以它们实际上并不是互相独立的,附着于动词就意味着不能附着于名词了,反之亦然。但由例句(8.18)中我们可知 VA_p 和 NA_p 的定义允许两者都是真的。独立性假设的优点在于分别计算两个变量的经验估计要比得到它们的联合分布的估计容易一些。下面将会看到怎样从未标注语料库中估计相关数值。

假设我们希望确定一个紧随宾语名词之后的介词短语的附着问题。使用式(8.20),通过计算 $NA_p = 1$ 的概率即可计算附着的估计值。

$$\begin{aligned} P(\text{Attach}(p) = n | v, n) &= P(VA_p = 0 \vee VA_p = 1 | v) \times P(NA_p = 1 | n) \\ &= 1.0 \times P(NA_p = 1 | n) \\ &= P(NA_p = 1 | n) \end{aligned}$$

由于在句子中会有其他的一些介词短语来修饰动词,所以我们不需要考虑 $VA_p = 1$ 或 $VA_p = 0$, 它们并不影响紧随名词之后的介词短语附着关系的确定。

为了证明 $VA_p = 1$ 和 $NA_p = 1$ 不能使 $\text{Attach}(p) = v$ 为真,我们考虑下面两个前提条件。第一,必须有两个以介词 p 开头的介词短语。这是因为任何一个给定的介词短语(PP)都只能附着于一个短语,或者是动词,或者是名词。第二,这两个介词短语中的第一个必须附着于名词,第二个附着于动词。这是因为如果还有其他的附着关系,我们得到的句法结构中就会出现交

叉括号(crossing brackets)的情况。由此得出结论, $VA_p = 1$ 和 $NA_p = 1$ 的情况暗示了第一个以 p 开头的介词短语附着于名词, 不附着于动词, 即 $\text{Attach}(p) = n$ 。因此 $\text{Attach}(p) \neq v$ 在这种情况下成立。

相反, 由于在短语结构树中不能有交叉点, 如果第一个以前置词 p 开头的介词短语附着于动词, 那么一定有 $VA_p = 1$ 和 $NA_p = 0$ 。由式(8.20)我们得到:

$$\begin{aligned} P(\text{Attach}(p) = v | v, n) &= P(VA_p = 1, NA_p = 0 | v, n) \\ &= P(VA_p = 1 | v) P(NA_p = 0 | n) \end{aligned}$$

通过似然比 λ 我们可以评价 $P(\text{Attach}(p) = v)$ 和 $P(\text{Attach}(p) = n)$ 。

$$\begin{aligned} \lambda(v, n, p) &= \log_2 \frac{P(\text{Attach}(p) = v | v, n)}{P(\text{Attach}(p) = n | v, n)} \\ &= \log_2 \frac{P(VA_p = 1 | v) P(NA_p = 0 | n)}{P(NA_p = 1 | n)} \end{aligned} \quad (8.21)$$

我们选择了动词附着关系作为 λ 的正值, 名词附着关系作为 λ 的负值。对于 λ 接近 0 的情况也可以做出判断, 但这时有一定的错误率。

在式(8.21)中我们用到了 $P(VA_p = 1 | v)$ 和 $P(NA_p = 1 | n)$, 怎样估计它们的概率值呢? 最简单的方法是使用最大似然估计:

$$\begin{aligned} P(VA_p = 1 | v) &= \frac{C(v, p)}{C(v)} \\ P(NA_p = 1 | n) &= \frac{C(n, p)}{C(n)} \end{aligned}$$

在公式中, $C(v)$ 和 $C(n)$ 是语料中 v 和 n 出现的次数, $C(v, p)$ 和 $C(n, p)$ 分别是 p 附着于 v 和 p 附着于 n 的次数。剩下的困难就是从未标注语料中确定附着关系的数量了。在一些句子中这些附着关系很明显。

- a. The road *to London* is long and winding.
- b. She sent him *into the nursery* to gather up his toys.

(8.22)

在例句(8.22a)中, 以斜体书写的介词短语一定附着于名词, 因为在名词之前没有动词了。在例句(8.22b)中, 以斜体表示的介词短语一定附着于动词, 因为它附着于代词(*him*)是不可能的。所以, 根据这两个句子, 我们可以把 $C(\text{road}, \text{to})$ 和 $C(\text{send}, \text{into})$ 的计数值分别加 1。但是, 很多句子都存在歧义问题, 所以我们希望能够采用一种自动化的过程来解决附着歧义问题。

Hindle and Rooth(1993)设计了一个启发算法, 用来从未标注数据中确定 $C(v, p)$ 和 $C(n, p)$, 从本质上讲包含以下三个步骤。

1. 通过计算所有的非歧义情况(如例句(8.22a)和例句(8.22b))构建初始模型。
2. 把这个初始模型应用到所有歧义情况中, 通过比较 λ 和预先定义的阈值的大小, 做出相应的附着关系的判断。例如, $\lambda > 2.0$ 表示动词附着关系, $\lambda < -2.0$ 表示名词附着关系。
3. 在两种附着关系计数中均匀地分配其余的歧义情况。也就是说, 对于每一个歧义情

况, $C(v, p)$ 和 $C(n, p)$ 各增加 0.5。

重新考虑例句(8.15a)作为例句(8.23), 可以示例怎样使用上述方法(Hindle and Rooth 1993: 109~110)。

Moscow sent more than 100 000 soldiers into Afghanistan ... (8.23)

首先, 估计似然率需要的两个概率值。统计数据来自于 Hindle and Rooth(1993)的测试语料库。

$$\begin{aligned} P(VA_{into} = 1|send) &= \frac{C(send, into)}{C(send)} = \frac{86}{1742.5} \approx 0.049 \\ P(NA_{into} = 1|soldiers) &= \frac{C(soldiers, into)}{C(soldiers)} = \frac{1}{1478} \approx 0.0007 \end{aligned}$$

公式中之所以出现小数计数, 是因为启发算法中把最难解决的歧义问题均匀划分到动词和名词计数中, 前面我们也提到过这一点。我们也可以得到:

$$P(NA_{into} = 0|soldiers) = 1 - P(NA_{into} = 1|soldiers) \approx 0.9993 \quad (8.24)$$

把这些数据代入式(8.21), 可以得到下面的似然比:

$$\lambda(send, soldiers, into) \approx \log_2 \frac{0.049 \times 0.9993}{0.0007} \approx 6.13$$

所以, 动词附着关系是最可能的(是名词附着可能性的 $2^{6.13} \approx 70$ 倍), 而且在这种情况下也是正确的预测。通常, 这种方法的准确率可以达到 80% 左右(Hindle and Rooth 1993: 115)。如果仅仅对 λ 值超过某个阈值的情况做出判断, 那么我们可以得到一个更高的精确率, 但是往往召回率较低。例如, Hindle and Rooth(1993)得到了 91.7% 的精确率和 55.2% 的召回率, $\lambda = 3.0$ 。

8.3.2 介词短语附着的一般评论

早期关于句法分析的心理语言学文献大多使用结构启发(structural heuristics)的方法来处理歧义问题, 但是这些方法都不能处理介词短语附着的歧义问题。对于同样的词类序列, 有时某一个句法分析结构是正确的, 有时另一个句法分析结构是正确的。正如 Ford et al. (1982)提出的观点, 词汇倾向性在这里起着很关键的作用。

如果使用模型来解决歧义问题, 主要有以下几个主要的限制: 一是我们仅仅考虑了前置词和它要附着的名词和动词。有时其他信息也是重要的(研究显示, 当看到比 v, n, p 更多的信息时, 人类判断附着歧义的准确率可以提高大约 5%)。在实践中, 在类似例句(8.25)的句子中, 包含在介词短语中的名词短语的中心词的识别是关键。

- a. I examined the man with a stethoscope
 - b. I examined the man with a broken leg
- (8.25)

但是, 其他的信息也可能是很重要的。例如, Hindle and Rooth(1993)注意到在名词前出现形容词最高级就很可能是名词短语附着(基于所用的数据)。这个条件可能是被忽略了, 因为形容词最高级出现的次数并不多。但是, 似然比方法的优点就是可以把其他的一些特征结合到自己的特征集中(假设这些特征是独立的)。在识别介词短语中名词短语的中心词的应用中, 其他很多研究都使用了不同的特征(Resnik and Hearst 1993; Brill and Resnik 1994; Ratnaparkhi et al. 1994; Zavrel et al. 1997; Ratnaparkhi 1998)。Franz(1996)已经能够把大量的特征包含到一个

对数线性模型中,但是也付出了一定的代价,即把最基本的关联强度参数都归结为类别参数^①。

第二个主要的限制是,Hindle and Rooth(1993)仅仅考虑了一个介词短语紧随充当宾语的名词短语之后的情况,而这个名词短语修饰前面的名词或者动词。但是,在实际的语料中,介词短语附着问题可能出现的情况要多得多。Gibson and Pearlmutter(1994)认为,由于对特殊情况的过度关注,心理语言学的研究已经被引入歧途了。如果一个介词短语和宾语名词之间被另一个介词短语隔开,那么该介词短语可以修饰它前面的介词短语中的任何一个名词,也可以修饰宾语名词或者前面的动词。图 8.2 中显示了一个在文本中出现的非常复杂的附着关系模式。此外,在复杂的句子中,介词短语可能不仅仅修饰前面邻近的动词,而且有可能修饰更前面的动词。Franz(1997)的论文中有更进一步的讨论。

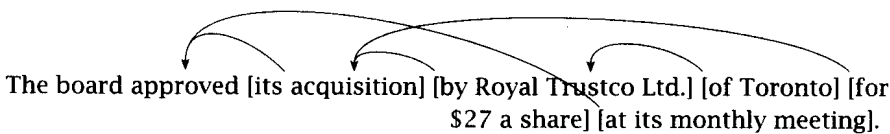


图 8.2 一个复杂句子中的附着关系

其他的附着问题

除了介词短语之外,附着歧义问题也存在于状语和分词短语,以及子句和复合名词(noun compounds)中。如何界定句法分析中的并列关系的范围,与附着问题的确定是很相似的,但是在这里我们不对这些问题进行更深一步的论述。

一个包含 3 个以上名词的名词短语或者是左分支(left-branch)结构[[N N]N],或者是右分支(right-branch)结构[N[N N]]。例如,door bell manufacturer 是左分支的: [[door bell]manufacturer]。它表示 door bell 的 manufacturer,而不是附着于 door 的 bell manufacturer。短语 woman aid worker 是一个右分支的例子: [woman[aid worker]]。这个短语暗指一个 aid worker 是女性。左分支的情况粗略对应于介词短语附着动词的情况([V N P]),右分支的情况对应于介词短语附着名词的情况([V[N P]])。

我们可以把应用在介词短语上的方法直接应用于复合名词。但是数据稀疏问题对复合名词问题更加严重,因为介词是高频词,而大部分名词则不是。因此,在有附着问题的复合名词中,可以使用基于词类的语义一般化方法并结合附着信息的方法。参见 Lauer(1995a)解决上述问题的方法。在下一节会讨论另一个基于类一般化的例子。

附着歧义问题的最后一个问题,我们注意到了大量的前置词短语表现出了“不确定”的附着关系(Hindle and Rooth 1993: 112)。考虑例句(8.26)中的介词短语 *with them*。

We have not signed a settlement agreement *with them*. (8.26)

注意,当我们和某人签署一个协议时,绝大部分情况下这个协议也是针对这个和我们签协议的人的,那么这个介词短语附着于动词还是名词就不清楚了。是否这样说更合适:例句(8.26)中的介词短语 *with them* 既附着于动词也附着于名词。Lauer(1995a)发现复合名词中有很大一部分

^① 具体情况请读者参阅相关的论文。——译者注

分也有这样的附着不确定问题。这个例子可能已经超出了统计自然语言处理工作要解决的问题范畴。在 Hindle and Rooth(1993)的研究之前,计算语言学家们通常还没有意识到附着不确定问题的广泛性,虽然 Church and Patil(1982)给出了一些反例。

在了解了不确定问题的事实之后,我们可以说在不确定例子中怎样的附着关系是无关紧要的。但是这个现象也驱使我们探寻新的方法来确定介词短语对句子所表达含义的贡献。而且,附着不确定现象也说明,正如目前的句法分析所限定的,要求介词短语的意思通过名词或动词短语来表示的方法可能是不恰当的。

8.4 选择倾向

大部分动词更倾向于有特定的论元类型。这个规律就是选择倾向或称选择约束。例如,动词 *eat* 的宾语倾向于食物类, *think* 的主语倾向于人, *bark* 的主语倾向于动物“狗”。这些论元约束和我们先前考虑的句法限制是类似的。例如,宾语的子范畴、介词短语和不定式。使用术语“倾向”(preference)而不用术语“规则”(rule)是因为这种“倾向”属性在比喻修辞和其他扩展意思中可以被忽略。例如,我们可以说吃(eat)一个非食物的对象, *eating one's word* 或 *fear eat the soul*。

在统计自然语言处理中选择倾向的获取是相当重要的。如果我们的电子词典中缺少某个词(比如, *durian*),那么可以从选择约束中得到它的部分意思。在例句(8.27)的情况下,我们就可以推测出 *durian* 是一种食物。

Susan had never eaten a fresh durian before. (8.27)

选择倾向的另一个重要用法是用于排列一个句子可能的句法分析结果。在排列中我们会倾向于某一类句法分析,根据选择倾向概念,在这个句法分析中的动词具有所谓的某种“自然”论元,从而可以赋予这类句法分析结果较高的分值。基于选择倾向,在自动语言处理中对一个句子的语义合适度的打分机制比完全理解句子的含义要更符合实际。这是因为,在选择倾向中的语义规律非常强,而且由于在动词和它的论元之间有很强的句法限制,我们可以很容易地从语料库中获取这些语义限制信息。

我们现在介绍一个由 Resnik(1993, 1996)提出的选择倾向模型。这个模型可以应用到任何词类的语义约束中:动词和主语、动词和直接宾语、动词和介词短语、形容词和名词、名词和名词(在名词-名词组合中)。但是我们在这里只考虑“名词和直接宾语”的情况,也就是说,动词选择直接宾语名词短语的语义限制。

这个模型使用两个概念形式化了选择倾向性:选择倾向强度(selectional preference strength)和选择关联(selectional association)。选择倾向强度度量了动词约束它的直接宾语的强度。它被定义为直接宾语的先验分布(普通动词直接宾语的分布)和我们所求的动词直接宾语的分布之间的相对熵(KL divergence)。

我们使用两个假设来简化这个模型。第一,只考虑直接宾语的中心词(例如, *Susan eat the green apple* 中的 *apple*),因为中心词是确定名词短语是否和动词匹配的关键部分。第二,不处理单独的名词,而是考虑名词类。通常,基于类的模型促进了模型的一般化,并且简化了参数估计。根据这些假设,我们可以定义选择倾向参数如下:

$$S(v) = D(P(C|v) \| P(C)) = \sum_c P(c|v) \log \frac{P(c|v)}{P(c)} \quad (8.28)$$

其中 $P(C)$ 表示名词类的总体概率分布, $P(C|v)$ 表示名词类作为动词 v 的直接宾语的概率分布。从任何一个具有名词分类信息的词典资源中可以获得名词的类别信息。Resnik (1996) 使用的是词网 (WordNet)。

基于选择倾向强度, 我们可以定义动词 v 和词类 c 之间的选择关联 (selectional association) 如下:

$$A(v, c) = \frac{P(c|v) \log \frac{P(c|v)}{P(c)}}{S(v)} \quad (8.29)$$

也就是说, 动词和词类之间的关联度被定义为一个比值: $P(c|v) \log P(c|v)/P(c)$ 和总的选择倾向强度 $S(v)$ 之间的比值。

最后我们还需要一个规则来指定关联强度和名词之间的关系 (而不是名词类)。如果名词 n 只属于某一类别 c , 那么我们只需简单地定义 $A(v, n) \stackrel{\text{def}}{=} A(v, c)$; 如果名词属于很多类, 那么它的关联强度定义为它的所有类别中关联强度最大的值。

$$A(v, n) = \max_{c \in \text{classes}(n)} A(v, c) \quad (8.30)$$

例如, 例句 (8.31) 中的名词 *chair* 就有几个类别, 因为它是一词多义的 (或者说是歧义的)。

$$\text{Susan interrupted the chair.} \quad (8.31)$$

在 *chair* 的情况中, 有两个候选类 *furniture* 和 *people* (后者的意思是 *chairperson*)。把 $A(v, n)$ 和最大的 $A(v, c)$ 等同起来可以对名词进行消歧。在例句 (8.31) 中, 我们把关联强度 $A(\text{interrupt}, \text{chair})$ 建立在基于 *people* 类的基础上, 因为打断一个人说话比打断一件家具更常用, 即:

$$A(\text{interrupt}, \text{people}) \gg A(\text{interrupt}, \text{furniture})$$

因此:

$$\begin{aligned} A(\text{interrupt}, \text{chair}) &= \max_{c \in \text{classes}(\text{chair})} A(\text{interrupt}, c) \\ &= A(\text{interrupt}, \text{people}) \end{aligned}$$

所以我们就在确定 *interrupt* 和 *chair* 的关联强度时, 顺便也对 *chair* 进行了消歧。

表 8.5 中的假设数据 (基于 (Resnik 1996: 139)) 可以作为模型进一步解释。表中显示了作为宾语的名词短语在名词分类 (假设只有显示的四个分类) 上的先验分布和对三个动词的后验分布。动词 *eat* 最有可能以 *food* 作为论元; *see* 的分布几乎和先验分布一样, 因为任何一种物体都可以被看见; *find* 在前三个类中有一致的分布, 但是“不喜欢”action, 因为 action 不是一个可以被发现的实体。

表 8.5 选择倾向强度 (SPS)。对于四个名词分类, 三个动词的论元分布和选择倾向强度 (基于假设数据)

名词类 c	$P(c)$	$P(c \text{eat})$	$P(c \text{see})$	$P(c \text{find})$
people	0.25	0.01	0.25	0.33
furniture	0.25	0.01	0.25	0.33
food	0.25	0.97	0.25	0.33
action	0.25	0.01	0.25	0.01
SPS $S(v)$		1.76	0.00	0.35

三个动词的选择倾向强度显示在最后的 SPS 行。这些数字和我们对这三个动词的直观感觉很一致: *eat* 相对于它要带的论元有很强的倾向性, *find* 次之, *see* 没有选择倾向性(至少在我們的假设数据中没有倾向性)。注意, 这里可以对 SPS 的含义进行清晰直观的解释, 它是我們学习动词之后论元的信息增益量。在 *eat* 的例子中, SPS 是 1.76, 几乎等于以 2 为基的对数值 2。正好是我们要进行的四(people, furniture, food, action)选一的以 2 为基的对数运算, 即 *eat* 选择了 food 类(以 2 为基的对数可以用来计算 SPS 和联合强度)。

计算动词和名词类之间的联合强度时, 我们发现 *eat* 非常倾向于 food 类, 见式(8.32), 反之 *find* 非常不倾向于 action 类, 见式(8.33)。这个例子显示模型除了公式化选择倾向强度(正数)之外, 还可以公式化非选择倾向强度(负数)。

$$A(\text{eat}, \text{food}) = 1.08 \quad (8.32)$$

$$A(\text{find}, \text{action}) = -0.13 \quad (8.33)$$

see 和其他四个名词类别之间的关联强度都为零, 直观感觉上 *see* 对它可能的语义对象没有强的限制。

剩下的问题是如何估计名词类别 c 作为给定动词 v 的直接宾语的概率: $P(c|v) = \frac{P(v, c)}{P(v)}$ 。 $P(v)$ 的最大似然估计是 $C(v)/\sum_v C(v')$, 即 v 对所有动词的相对频率。Resnik (1996) 使用下式估计 $P(v, c)$:

$$P(v, c) = \frac{1}{N} \sum_{n \in \text{words}(c)} \frac{1}{|\text{classes}(n)|} C(v, n) \quad (8.34)$$

在上面的公式中, N 是语料中动词-宾语对的总数量, $\text{words}(c)$ 表示名词类 c 中所有名词的集合, $\text{classes}(n)$ 表示包含名词 n 的名词类的数目, $C(v, n)$ 表示动词 v 和以 n 为中心词的宾语名词短语对的数目。这种估计 $P(v, c)$ 的方法避开了名词消歧的问题。如果一个名词的两个类别 c_1 和 c_2 都和 v 同现, 那么我们可以把它的一半出现次数指定到 $P(v, c_1)$, 另一半指定到 $P(v, c_2)$ 。

迄今为止, 我们仅仅给出了结构化的例子。表 8.6 中给出了一些真实的数据, 它们是 Resnik 在 Brown 语料库上的实验数据 (Resnik 1996: 142)。这些动词和名词都取自相关的心理语言学研究 (Holmes et al. 1989)。表中左边和右边的名词分别是“典型”和“非典型”宾语。对于大部分动词来说, 关联强度能够准确地预测哪一个宾语是典型的。例如, 可以准确预测出 *friend* 作为 *see* 的宾语的可能性要比 *method* 大。我们注意到, 模型的大部分错误是由于模型的消歧形式引起的, 模型通过在名词可能的类之间选择最高的关联强度进行消歧(参见先前讨论的 *chair* 的例子)。如果某个名词不属于典型宾语类别, 但在它的各种含义中存在一个含义使它可以合理地充当宾语, 即使这个含义非常罕见, 按照关联强度大小我们也把它划归为典型宾语。以 *Hear* 为例, 单词 *story* 和 *issue* 都属于 communication 类。*issue* 的这种含义属于前面所提到的所谓罕见解释, 但是模型选择它作为动词 *hear* 的宾语, 因为由它充当动词的宾语会更加合理。

表 8.6 使用关联强度区别一个动词的典型和非典型宾语。表的左半部分是典型宾语,右半部分是非典型宾语。在大多数情况下,关联强度 $A(v, n)$ 是宾语典型性的一个很好的指示器

动词 v	名词 n	$A(v, n)$	类	名词 n	$A(v, n)$	类
<i>answer</i>	<i>request</i>	4.49	speech act	<i>tragedy</i>	3.88	communication
<i>find</i>	<i>label</i>	1.10	abstraction	<i>fever</i>	0.22	psych. feature
<i>hear</i>	<i>story</i>	1.89	communication	<i>issue</i>	1.89	communication
<i>remember</i>	<i>reply</i>	1.31	statement	<i>smoke</i>	0.20	article of commerce
<i>repeat</i>	<i>comment</i>	1.23	communication	<i>journal</i>	1.23	communication
<i>read</i>	<i>article</i>	6.80	writing	<i>fashion</i>	-0.20	activity
<i>see</i>	<i>friend</i>	5.79	entity	<i>method</i>	-0.01	method
<i>write</i>	<i>letter</i>	7.26	writing	<i>market</i>	0.00	commerce

除了选择倾向的特殊问题外, Resnik 也研究了模型预测动词是否有所谓的隐式宾语交替 (implicit object alternation) 问题, 或称非确定宾语交替 (unspecified object alternation), 参见 Levin (1993: 33)。例如例句(8.35a)和例句(8.35b)中的宾语交替问题。动词 *eat* 的语义对象可以在明确化的宾语和省略的宾语之间选择。

- (8.35)
- a. Mike ate the cake.
- b. Mike ate.

对于这种现象 Resnik 是这样解释的: 动词对宾语的限制越多, 它就越有可能允许隐式宾语的结构。对于像 *eat* 这样带有很强选择倾向的动词, 直观的感觉是只要知道了动词给予的大量信息, 它的直接宾语就不用再提及了。Resnik 发现, 选择倾向强度对动词的隐式宾语交替是否出现可以起到很好的指示作用。

我们现在可以看到 Resnik 的模型定义选择倾向强度 (SPS) 作为主要概念, 并且在它的基础上计算关联强度的原因了。SPS 可以被看成是一个基本现象, 除了关联强度之外它还可以解释隐式宾语的出现。

另一种方法是直接定义关联强度为 $P(c|v)$ 或者 $P(n|v)$ (如果不希望有中间的名词类表示)。计算 $P(n|v)$ 的方法包括了分布聚类 (第 14 章描写的 Pereira et al. (1993) 的工作) 和计算名词相似性的方法。如果名词相似性的方法是可用的, 那么可以在相似于 n 的名词分布中计算 $P(n|v)$ 。下一节会更详细地讨论这种方法。

8.5 语义相似性

词汇获取的最高目标是词义的获取。如果我们能够自动获取词义, 那么统计自然语言处理中的许多任务 (例如文本理解和信息检索处理) 的结果将会有很大的改进。遗憾的是, 怎样去表述词义并为自动系统所用还是一个尚未解决的问题。因此, 大部分词汇语义属性获取的工作就集中在了语义相似性上。因为自动获得一个新词相对于已知词汇的语义相似性要比获取这个新词的确切意思容易得多。

虽然有自身的限制, 但是语义相似性还是为我们提供了一个很有用的评价方法。假设语义相似的词语有某些相似的语言行为, 我们经常使用语义相似性对文本进行一般化 (generalization) 处理。例如, 对于上一节讨论的选择歧义问题, 假设我们希望知道在例句(8.37) (参见前

面的例句(8.27)中 *durian* 作为 *eat* 的语义对象是否合适

Susan had never eaten a fresh durian before. (8.37)

并且假设我们没有任何关于 *durian* 的语义信息,但是知道它和 *apple*, *banana* 和 *mango* 是语义相似的,这些词都非常适合 *eat* 的选择倾向。那么,我们就可以把 *apple*, *banana* 和 *mango* 的行为特点相似地推广到 *durian*,而且可以假设 *durian* 也是 *eat* 的一个合适的语义对象。通过灵活地使用这种策略,我们可以把 *durian* 的语义相似性仅仅建立在语义最近似的词汇上(例如 *mango*),也可以把它建立在几种最近似语义词汇的组合上,这种组合与 *durian* 的关系可以通过对每个词汇和 *durian* 近似程度的加权得到。

相似性一般化(similarity-based generalization)和类一般化(class-based generalization)的处理在一定程度上讲有些近似。在相似性一般化方法中,我们考虑的只是一个词的最相似的邻近词。而在类一般化中,我们要考虑和一个词有关联的一类词的特性(参见习题 8.20)。

语义相似性在信息检索中还被用来做查询扩展(query expansion)。用户自己描述的信息需求可能和它要找的文本中相应的术语有很大出入。举例说明,当一个用户想要找一些关于俄罗斯航天任务的文章时,他使用 *astronaut* 作为查询,那么查询扩展系统可以基于 *astronaut* 和 *cosmonaut* 之间的语义相似性把查询扩展到 *cosmonaut*。

语义相似性的另一个应用是所谓的 K 近邻法(K nearest neighbors)分类(参见 16.4 节)。我们首先需要有一个标有类别属性的训练集。训练集里的元素可以是词汇,类别可以是新闻报道中的主题范畴(financial, agriculture, politics 等)。在 K 近邻法分类中,根据它的 k 个近邻可以作为一个新的元素指定一个最可能的类别。

在我们深入研究怎样获取语义相似性之前,首先要注意到语义相似性还没有一个直观和清晰的概念。在某种程度上说,语义相似性是近义词的扩展,比如 *dwelling*/*abode* 这样的近义词对。通常,语义相似性是指来自于相同语义领域和主题(semantic domain and topic)的两个词。基于这种理解,如果词汇涉及到的事物在现实世界中很可能共同出现,例如 *doctor*, *nurse*, *fever* 和 *intravenous*,那么这些词汇就是相似的,但是上述词汇可能涉及的是非常不同的主体,甚至它们的句法范畴都是不同的。

Miller and Charles(1991)希望进一步具体化语义相似的概念,认为语义相似的判断可以通过上下文可换性(contextual interchangeability)的程度来解释,也就是说,用一个词在上下文中可以被另一个词代替的程度来度量语义相似性。

所有的语义相似概念都含有歧义问题。如果一个词和一个歧义词的一个语义相似,那么它和这个词的其他语义基本上是不相似的。例如,单词 *litigation* 和 *suit* 的法律语义是相似的,但是它和 *clothes* 语义是不相似的。当把语义相似性的概念应用到歧义词时,语义相似通常意味着和一个适合的语义相似。

8.5.1 向量空间度量方法

我们知道很多度量语义相似性的方法实际上被映射为向量相似性的度量,即需要计算语义相似性的两个词可以转换为多维空间中的向量形式。图 8.3,图 8.4 和图 8.5 给出了多维空间的例子(参看图 15.5)。

	cosmonaut	astronaut	moon	car	truck
d_1	1	0	1	1	0
d_2	0	1	1	0	0
d_3	1	0	0	0	0
d_4	0	0	0	1	1
d_5	0	0	0	1	0
d_6	0	0	0	0	1

图 8.3 文本-词汇矩阵 A

	cosmonaut	astronaut	moon	car	truck
cosmonaut	2	0	1	1	0
astronaut	0	1	1	0	0
moon	1	1	2	1	0
car	1	0	1	3	1
truck	0	0	0	1	2

图 8.4 词汇-词汇矩阵 B

	cosmonaut	astronaut	moon	car	truck
Soviet	1	0	0	1	1
American	0	1	0	1	1
spacewalking	1	1	0	0	0
red	0	0	0	1	1
full	0	0	1	0	0
old	0	0	0	1	1

图 8.5 中心词-修饰词矩阵 C 。最上边一行是名词或名词短语的中心词,最左边一列是修饰它们的修饰词

图 8.3 中的矩阵表示了文档空间中的词向量。元素 a_{ij} 表示词 j 在文档 i 中出现的次数。词语是否相似取决于它们出现于相同文档中的次数。在文档空间中, *cosmonaut* 和 *astronaut* 是不相似的(没有出现在相同的文档中); *truck* 和 *car* 是相似的,因为它们在一个文档 d_4 中同现。

图 8.4 表示了词空间中的词向量。元素 b_{ij} 表示词 j 和词 i 的同现次数。同现可以根据文章、段落或其他文章长度单位来定义。词语是否相似取决于它们和相同词汇同现的次数。所以,在这种情况下, *cosmonaut* 和 *astronaut* 比在上面一个空间中更相似,因为它们都和 *moon* 同现。

图 8.4 中的同现是根据图 8.3 中的文档来定义的。也就是说,有下面的等式成立: $B = A^T A$ 。在这里, T 表示转置(transpose),它是对行列的交换,所以有 $X_j^T = X_i$ 。

图 8.5 表示了修饰词空间(modifier space)中的名词向量(或名词短语的中心词)。元素 c_{ij} 表示了中心词 j 被修饰词 i 修饰的次数。中心词是否相似取决于它们被相同的修饰词修饰的次数。在这个空间中, *cosmonaut* 和 *astronaut* 是相似的。而且,与图 8.3 中的文档空间和图 8.4 中的词空间不同的是, *moon* 和 *cosmonaut*, *astronaut* 是不相似的。这些区别表明,不同空间体现了不同类型的语义相似性。通过在文档和词空间中使用的同现信息可以得到典型的相似性(typical similarity)(相似词属于相同的主题领域)。中心词-修饰词信息更细致一些。比如,虽

然 *astronaut* 和 *moon* 属于相同的主题(space exploration),但是很明显它们表示的实体具有非常不同的属性(一个是人,一个是天体)。不同的属性就会对应不同的修饰词,这也就解释了这两个词在中心词-修饰词矩阵中不相似的原因^①。

对于这三个矩阵,我们还可以考虑它们的行间相似性和列间相似性。使用这种方法重新定义矩阵,矩阵 *A* 定义了文档之间的相似性,这也是定义文档之间相似性和文档与信息检索问题之间相似性的标准方法。矩阵 *C* 定义了修饰词之间的相似性。例如, *red* 和 *old* 就是相似的(它们都可以修饰 *car* 和 *truck*),这表明它们可以修饰相同类型的名词。矩阵 *B* 是对称矩阵,所以它的行间相似性和列间相似性是一样的。

迄今为止,我们已经有了一个向量相似性的直观概念。表 8.7 定义的几种度量方法更加准确地表明了这个概念(改编自(van Rijsbergen 1979: 39))。首先,我们仅仅考虑的是二元向量,也就是说,向量中每一维的值是 0 或 1。描述二元向量最简单的方法就是维度集合。例如,图 8.5 中 *cosmonaut* 的向量是 {Soviet, spacewalking}。有了维度集合之后,我们就可以使用表 8.7 中的集合操作来计算相似度。

表 8.7 二元向量的相似性度量

相似性测量	定义
匹配系数	$ X \cap Y $
Dice 系数	$\frac{2 X \cap Y }{ X + Y }$
Jaccard (或 Tanimoto) 系数	$\frac{ X \cap Y }{ X \cup Y }$
交迭系数	$\frac{ X \cap Y }{\min(X , Y)}$
cosine	$\frac{ X \cap Y }{\sqrt{ X \times Y }}$

表 8.7 中第一个相似性度量,即匹配系数(matching coefficient),简单地计数了两个向量都非零的维的数目。与其他度量方法不同的是,它忽略了向量的长度和每一个向量非零维数的影响^②。

Dice 系数(Dice coefficient)对向量长度进行了处理,即将两个向量总的非零维的数目作为公式的分母。在定义中乘以 2 的目的是为了使度量值限制在 0.0 到 1.0 的范围内,度量值为 1.0 表明两个向量相等。

小数目共享元对 Jaccard 系数的不利影响要大于 Dice 系数。这两个度量方法的度量值都在 0.0(无交迭)和 1.0(完全交迭)之间,但是在交迭程度较小的情况下,由 Jaccard 系数得出的值较低。例如,有两个具有 10 个非零元的向量,它们只有一个非零元是重叠的,因此可以得到 Dice 值 $2 \times 1 / (10 + 10) = 0.1$ 和 Jaccard 值 $1 / (10 + 10 - 1) \approx 0.05$ 。比较而言, Jaccard 系数在化学中更常用,它可以度量两个化合物之间的相似性(Willett and Winterman 1986)。

交迭系数(overlap coefficient)可以度量互相之间的包含性。如果第一个向量中的每一非零维对应于第二个向量也是非零的(反之亦然),换句话说就是 $X \subseteq Y$ 或者 $Y \subseteq X$,那么交迭系数为 1。

① 参见 Grefenstette(1996)和 Schütze and Pedersen (1997)的论述。

② 这种方法非常适合于反映相似性判断的可信度。Hindle(1990)推荐使用这种方法来判断名词相似性。

在向量中含有相同数目的非零维元素时,余弦(cosine)度量和 Dice 系数度量是一致的(参见习题 8.24),但是 Dice 度量在向量维数相差比较大的情况下效果不好。例如,如果我们比较一维非零元向量和 1000 维非零元向量,假设它们的共享维数为 1,那么我们得到的 Dice 系数是 $2 \times 1/(1+1000) \approx 0.002$,而余弦度量是 $1/\sqrt{1000 \times 1} \approx 0.03$ 。余弦的特性在统计自然语言处理中是非常重要的,因为我们会经常比较不同数据量上的词语或者对象,而且不希望因为数据量的差异而断定它们是不相似的。

到目前为止,我们考虑的都是二元向量,但是二元向量在每一维上只有很少的信息量可以供我们使用。对于语言对象,更有效的表达方式应该是实值向量空间(real-valued vector space)。在这里由于篇幅所限,我们就不给出线性代数的系统介绍了,只是简要地回顾一下向量空间的基本概念。一个 n 维的实值向量 $\vec{x}$ 是 n 个实数的序列,在这里 x_i 表示向量 $\vec{x}$ 的第 i 个元素(第 i 维的值)。一个向量的元素可以写成如下形式:

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad (8.38)$$

但是,在段落中我们有时也会把向量写成水平的表示形式。使用 $\mathbb{R}^n$ 表示 n 维实值向量的向量空间,因此有 $\vec{x} \in \mathbb{R}^n$ 。在欧几里得向量空间中,向量的长度定义如下:

$$|\vec{x}| = \sqrt{\sum_{i=1}^n x_i^2} \quad (8.39)$$

最后,定义两个向量之间的点积为: $\vec{x} \cdot \vec{y} = \sum_{i=1}^n x_i y_i$

我们上面介绍的度量二元向量的最后一个方法——余弦度量法,对于实值向量的度量也是很重要的。余弦法度量了空间中两个向量夹角的余弦,它的值从 1.0(两个向量同向)到 0.0(两个向量正交)再到 -1.0(两个向量反向)。

关于实值空间中的两个 n 维向量 $\vec{x}$ 和 $\vec{y}$,我们可以使用如下公式计算它们的余弦值:

$$\cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| |\vec{y}|} = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}} \quad (8.40)$$

这个定义实际上是余弦度量的另一种解释,即归一化相关系数(normalized correlation coefficient)。我们首先计算 x_i 和 y_i 的相关度,然后各自除以它们的欧氏距离。如果在欧氏空间中向量有单位长度,那么我们称该向量是归一化的(normalized):

$$|\vec{x}| = \sqrt{\sum_{i=1}^n x_i^2} = 1 \quad (8.41)$$

对于归一化向量,余弦值可以简化为点积的形式:

$$\cos(\vec{x}, \vec{y}) = \vec{x} \cdot \vec{y} \quad (8.42)$$

两个向量的欧氏距离度量了它们在向量空间中的距离远近:

$$|\vec{x} - \vec{y}| = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (8.43)$$

余弦度量存在一个有趣的特性,即如果使用归一化向量,那么余弦给出的相似性排列和欧氏距离给出的是一样的。也就是说,如果我们仅仅想知道两个对象中的哪一个和第三个对象更接近,那么对于归一化向量,余弦和欧氏距离的答案是一样的。下面的推导说明了余弦和欧氏距离给出的排列相同的原因:

$$\begin{aligned}
 (|\bar{x} - \bar{y}|)^2 &= \sum_{i=1}^n (x_i - y_i)^2 \\
 &= \sum_{i=1}^n x_i^2 - 2 \sum_{i=1}^n x_i y_i + \sum_{i=1}^n y_i^2 \\
 &= 1 - 2 \sum_{i=1}^n x_i y_i + 1 \\
 &= 2(1 - \bar{x} \cdot \bar{y})
 \end{aligned} \tag{8.44}$$

最后,余弦也可以被用来作为概率相似性的度量(Goldszmidt and Sahami 1998)。概率分布 $\{p_i\}$ 和 $\{q_i\}$ 首先被变换为 $\{\sqrt{p_i}\}$ 和 $\{\sqrt{q_i}\}$ 的形式,然后使用它们的余弦给出度量值 $D = \sum_{i=1}^n \sqrt{p_i q_i}$,这个度量可以被看成是两个分布的几何平均。

表 8.8 显示了我们计算的纽约时报语料库中一些单词的余弦相似性。和图 8.4 中的矩阵类似,我们编辑了一个 $20\,000 \times 1000$ 的矩阵。在行中选择了 20 000 个常用词,在列中选择了 1000 个常用词(在两种情况中都排除了前 100 个最常用的词)。我们使用对数加权方程 $f(x) = 1 + \log(x)$ (参见 15.2.2 节)代替了原始的同现计数。同现定义为两个词出现在彼此 25 个词的范围内。表中显示的是矩阵行间的余弦相似性。

表 8.8 使用余弦度量语义相似性。对于左边一列中的五个词,都给出了其最相似的词(右边),根据的就是词词同现矩阵中的余弦度量。例如, *sauce* 是 *garlic* 的最相似的词。*sauce* 和 *garlic* 向量之间的余弦值是 0.732

焦点词	最近邻							
<i>garlic</i>	<i>sauce</i>	0.732	<i>pepper</i>	0.728	<i>salt</i>	0.726	<i>cup</i>	0.726
<i>fallen</i>	<i>fell</i>	0.932	<i>decline</i>	0.931	<i>rise</i>	0.930	<i>drop</i>	0.929
<i>engineered</i>	<i>genetically</i>	0.758	<i>drugs</i>	0.688	<i>research</i>	0.687	<i>drug</i>	0.685
<i>Alfred</i>	<i>named</i>	0.814	<i>Robert</i>	0.809	<i>William</i>	0.808	<i>W</i>	0.808
<i>simple</i>	<i>something</i>	0.964	<i>things</i>	0.963	<i>You</i>	0.963	<i>always</i>	0.962

对于某些词对而言,词空间上的余弦是衡量相似性的一种有效方法。举例说明, *garlic* 周围的词通常在意意思上也和 *garlic* 相近,这一点对 *fallen* 也是一样的。但是,由于同现信息并不能反映词序和语法依存关系(比如说过去分词 *fallen* 和过去时 *fell* 就是最邻近的两个词),所以这种方法并不能反映语法差别。词汇 *engineered* 显示了相似性度量对语料库的依赖性,在纽约时报语料库中,它通常出现在和基因工程相关的上下文中,而在一个汽车杂志文章的语料中, *engineered* 的相邻词可能就会迥然不同。最后,我们利用单词 *Alfred* 和 *simple* 说明这种相似性度量的局限性。*Alfred* 的一些邻近词通常也是表示名字的单词,这些名字和 *Alfred* 之间是词性相似而不是语义相似。而 *simple* 的邻近词似乎是完全随机的,因为 *simple* 使用得非常频繁,并且分布在整个语料中,同现信息不能特征化这个词的语义。

我们所给出的例子都证明了向量空间作为一个具有代表性方法的优点:简洁性。在二维

或三维空间中很容易形象化向量,而且,认为指向同一方向的向量是相似的,这也非常直观。另外,向量空间度量法计算也比较简单。总之,向量空间度量具有简洁直观和计算简单的优点,因此它在信息检索中被长期使用,主要使用的是词汇-文档矩阵(Lesk 1969; Salton 1971a; Qiu and Frei 1993)。对词汇-词汇矩阵和修饰词-中心词矩阵的使用是在最近一些年(Grefenstette 1992b; Schütze 1992b)才兴起的。Grefenstette (1992a)和 Burgess and Lund (1997)中证明了基于向量的相似性度量和语义相似性心理学概念之间的内在联系,例如一个词可以在多大程度上给另一个词提供信息。

8.5.2 概率度量方法

基于向量空间的度量方法的问题是,除了余弦之外,它们的操作都是基于二值数据的(是或不是)。余弦是向量空间度量法中惟一可以处理数字信息的方法,但它也有自己的问题。余弦的计算需要在欧氏空间中,这是因为余弦被定义为三角形两边长度的比值,所以我们需要度量长度,即欧氏距离。但是,如果我们处理的向量是概率或者计数向量,那么欧氏空间并不是一个很好的选择,而且当计算语义相似性时,概率向量是我们最常用的表示形式。我们可以看到,在欧氏距离中概率 0.1 和 0.0 的距离与概率 0.9 和 1.0 的距离是一样的。但是,在第一种情况下,概率事件的差别是不可能与十分之一;相反,在第二种情况中,两个概率事件大约只有 10% 的差别。所以欧氏距离对正常分布的量是合适的,但是它不适合应用于概率和计数。

在图 8.3,图 8.4 和图 8.5 中的计数矩阵可以很容易地转化为条件概率矩阵,通过把每列的数据除以这一列的数据总和的方法即可得到概率数据(相当于使用最大似然估计)。例如,对于图 8.5 中的矩阵,数据(American, astronaut)转化为条件概率 $P(\text{American} | \text{astronaut}) = \frac{1}{2} = 0.5$ 。这样,语义相似性的问题就可以变为两个概率分布相似性(或不相似性)的问题了。

表 8.9 显示了概率分布中不相似性的三种度量方法,这些方法是由 Dagan et al. (1997b) 给出的。在 2.2.5 节中我们已经熟悉了相对熵(KL divergence)的概念。它度量了分布 q 近似于分布 p 的程度;或者更准确地说,当我们使用分布 q 代替真实分布 p 时,有多少信息会丢失。相对熵在实际应用中会遇到两个问题。第一,如果遇到 $q_i = 0$ 而 $p_i \neq 0$ 的情况(这种情况会经常出现,特别是当我们使用简化的最大似然估计时),相对熵的值为无穷大。第二,相对熵是不对称的,也就是说,通常 $D(p \| q) \neq D(q \| p)$ 。但是从直观上讲,语义相似性和其他大部分我们研究的相似性都应该是对称的,所以应该有以下式成立: $\text{sim}(p, q) = \text{sim}(q, p)$ ^①。

表 8.9 不同概率分布之间(不)相似性的度量

(不)相似性度量	定义
KL divergence	$D(p \ q) = \sum_i p_i \log \frac{p_i}{q_i}$
information radius (IRad)	$D(p \ \frac{p+q}{2}) + D(q \ \frac{p+q}{2})$
L_1 norm	$\sum_i p_i - q_i $

① 注意,在聚类中非对称性是有意义的,因为我们比较的是两个不同的实体,一个是我们要给它指定类别的单词,一个是要聚的类。聚类的问题是这个类在多大程度上表达这个词和严格意义上的语义相似性是不同的。参见(Pereira et al. 1993)。

表 8.9 中的第二种度量方法是信息半径 (information radius), Dagan et al. (1997b) 称之为 total divergence to the average。这种方法克服了上一方法的两个问题。它是对称的 ($\text{IRad}(p, q) = \text{IRad}(q, p)$), 而且它的值不会是无穷大, 因为 $\frac{(p_i + q_i)}{2} \neq 0$ 。IRad (信息半径) 的直观解释是, 它回答了这样的问题: 如果我们用两个词 (在通常情况下是随机变量) 的平均分布来表述这两个词, 将会有多少信息丢失? IRad 的值在 0 (同一分布) 到 $2\log 2$ (最大差别分布) (参见习题 8.26) 之间变化。一般假设 $0\log 0 = 0$ 。

第三种方法是 Dagan et al. (1997b) 提出的 L_1 范式 (或 Manhattan 范式)。这种方法也是对称的, 并且对任意的 p 和 q 都有明确的定义。我们可以把它看成是差异事件的期望比例, 也就是说, 在分布 p 和 q 中不同事件的比例。这是因为 $\frac{1}{2}L_1(p, q) = 1 - \sum_i \min(p_i, q_i)$, 而 $\sum_i \min(p_i, q_i)$ 是带有相同实验结果的期望比例^①

例如, 考虑下面由图 8.5 计算得到的条件概率:

$$\begin{aligned} p_1 &= P(\text{Soviet}|\text{cosmonaut}) = 0.5 \\ p_2 &= 0 \\ p_3 &= P(\text{spacewalking}|\text{cosmonaut}) = 0.5 \\ q_1 &= 0 \\ q_2 &= P(\text{American}|\text{astronaut}) = 0.5 \\ q_3 &= P(\text{spacewalking}|\text{astronaut}) = 0.5 \end{aligned}$$

可以得到:

$$\frac{1}{2}L_1(p, q) = 1 - \sum_i \min(p_i, q_i) = 1 - 0.5 = 0.5$$

所以, 如果我们考虑和 *cosmonaut* 以及 *astronaut* 一起出现的形容词集, 那么这两个词集交迭的期望是 0.5, 相当于 *spacewalking* 和每个名词同现的比例。

Dagan et al. (1997b) 在类似选择倾向问题的任务中比较了这三种非相似性度量方法 (KL, IRad 和 L_1)。在任务中, 不是考虑名词作为动词论元的合理性, 而是着眼于动词作为名词的谓语的合理性。例如, 在选择动词 *make* 和 *take* 时, 相似性度量可以确定 *make* 和 *plans* 连用 (*make plans*), *take* 和 *actions* 连用 (*take actions*)。

相似性度量可以用来计算条件概率 $P(\text{verb}|\text{noun})$ 。Dagan et al. (1997b) 使用它来衡量吻合度 (goodness of fit):

① 由下面的推导可以得到 $\frac{1}{2}L_1(p, q) = 1 - \sum_i \min(p_i, q_i)$:

$$\begin{aligned} L_1(p, q) &= \sum_i |p_i - q_i| \\ &= \sum_i [\max(p_i, q_i) - \min(p_i, q_i)] \\ &= \sum_i [(p_i + q_i - \min(p_i, q_i)) - \min(p_i, q_i)] \\ &= \sum_i p_i + \sum_i q_i - 2 \sum_i \min(p_i, q_i) \\ &= 2(1 - \sum_i \min(p_i, q_i)) \end{aligned}$$

注意, 由于 $\sum_i \min(p_i, q_i) \geq 0$, 所以 $0 \leq L_1(p, q) \leq 2$ 。

$$P_{\text{SIM}}(v|n) = \sum_{n' \in S(n)} \frac{W(n, n')}{N(n)} P(v|n') \quad (8.45)$$

公式中, v 是动词, n 是名词, $S(n)$ 是根据相似性度量得到的最类似于 n 的名词集^①。 $W(n, n')$ 是相似性度量, $N(n)$ 是规范化因子: $N(n) = \sum_{n'} W(n, n')$ 。

这个公式必须把非相似性度量(KL, IRad 和 L_1)转化为相似性度量 W 。使用如下三个转化公式:

$$W_{\text{KL}}(p, q) = 10^{-\beta D(p||q)} \quad (8.46)$$

$$W_{\text{IRad}}(p, q) = 10^{-\beta \text{IRad}(p||q)} \quad (8.47)$$

$$W_{L_1}(p, q) = (2 - L_1(p, q))^\beta \quad (8.48)$$

参数 β 可以根据最优性能调整。

Dagan et al. (1997b) 表明 IRad 始终好于其他两种方法, 因此建议一般情况下最好使用 IRad 方法。

作为结论, 我们简短地概括一下语义相似性和不相似性的度量方法。向量空间度量概念简洁, 并且产生的相似性值可以直接用来一般化文本, 但是它们缺乏计算方法的清晰解释。概率非相似性度量在理论上更加严谨, 但是它需要把非相似性转化为相似性才能用来一般化文本。这两种方法对于从语料中获取语义特性都是有益的, 它们通过使用相似性, 把已知词汇的知识转化到没有收入词典中的词汇上。

8.6 统计自然语言处理中词汇获取的作用

词汇获取在统计自然语言处理中扮演了关键的角色, 因为可利用的词汇资源总是很缺乏。

我们知道, 造成词汇资源缺乏的原因有很多。一个原因就是人工建立词汇资源的费用问题。对于很多类型的词汇信息, 专门的词典编撰者能够比自动程序收集到更准确、更全面的数据。但是, 由于费用的原因, 通常手工构造词典是不可行的。有人估计过, 如果从头开始创建一个词条, 需要的平均时间是半个小时 (Neff et al. 1993), 很明显这个时间依赖于词条的复杂性, 所以手工构建资源是非常昂贵的。

还有一种数据类型: 数字信息, 是人类(包括词典编撰者)很难收集的, 因此词汇获取中的数字信息不得不用机器自动完成, 甚至手工构建好的词汇资源也要使用这些数字特性。

而且, 很多词汇资源是根据人类的需求而设计的。数字信息越来越受重视, 是因为计算机无法处理传统词典中解释词条的上下文信息, 但是数字信息对于人类来说并不太重要。Mercer (1993) 恰当地形容了这一点: “我们通过一本双语词典无法学习一门新的语言。”例如, 非规则复数形式 *postmen* 在一些词典中没有被列为 *postman* 词条的特例, 因为很明显人类可以通过类比 *man* 的复数 *men* 得出 *postman* 的复数。对这类问题最好的解决方法就是通过自动方法增大手工资源的数量。

词汇获取之所以重要的主要原因在于语言的内在生产力。自然语言永远是在变化的, 它

^① 对于这个实验, $S(n)$ 选择的是名词集, 但是也可以把词集限于最接近的目标词。

在适应着变化的世界,它在不停地创造着新的名字和词来表示新事物、新的人和新的概念。词汇资源就不得不通过升级来和这些变化保持一致。相对来说某些类型的词更有可能演化出新词。例如,大部分文档都会涉及一些专有名词,这些词我们以前没有遇到过,但是人们很难经常创造助词和介词。新的名词和动词在文本中出现的概率也比较高。词典中包含的词也有词汇获取的需要,因为它们也会发展出新的语义或者新的句法模式。

我们怎样量化词汇信息呢? 这些信息是自动学习得到的,也可能是在词汇资源中已经包含的。对于比较粗略的评价,我们可以参考 Zipf 规则和其他的评价方法,这些方法估计了从未见过的词汇和文本中用过的词汇的比例,参见第6章, Baayen and Sproat (1996) 和 Youmans (1991)。

Sampson (1989)给出了更加细致的分析。Sampson 测试了一个接近 70 000 个词条的词典 (the OALD, Hornby 1974) 对 LOB 语料库中的一个 45 000 个词的子部分的覆盖率(数字没有不算成词)。他发现大约有 3% 的词次没有在词典中出现。考虑不同类型的词汇引起的覆盖问题是有益的。表 8.10 列出了 Sampson 发现的主要类型和一些例子。

表 8.10 出现在 LOB 语料库中却没有被 OLAD 词典覆盖的词类

覆盖问题的类别	实例
proper noun	<i>Caramello, Château-Chalon</i>
foreign word	<i>perestroika</i>
code	<i>R101</i>
mathematical object	x_1
non-standard English	<i>havin'</i>
abbreviation	<i>NLP</i>
hyphenated word	<i>non-examination</i>
hyphen omitted	<i>bedclothes</i>
negated adjective	<i>unassailable</i>
adverbs	<i>ritualistically</i>
technical vocabulary	<i>normoglycaemia</i>
plural of mass noun	<i>estimations</i>
other cases	<i>deglutition, don'ts, affinitizes (VBZ)</i>

一半以上的缺失词都是专有名词,另外一半是由表中的其他类型引起的。一些覆盖问题在大一些的词典中就不会遇到了(一些频繁出现的专有名词和像 *unassailable* 这样的名词)。但是,根据 Sampson 的发现,我们可以预期,即使使用一个非常大的词典,语料库中还是会有 1% ~ 2% 的词次没有被覆盖到。而且还要看到,这样的研究只能针对词典中存在的字符串,很难估计一个已知词有多大的可能性使用一个新的语义或句法结构。最后,这 1% ~ 2% 的未知词倾向于文档的最重要部分:在文章中刻画的人名或者一个新的科学现象的缩写。所以即使新词只占了文本的很小一部分,对其属性的表示也是极为重要的。

这些缺点将会持续很长一段时间,直到自然语言处理的研究者解决了对语言处理中词典和手工知识获取的限制。在早期的自然语言处理研究中,一个常用的策略就是专注于一个小的子领域,解决两个主要的基本问题:句法分析和知识表示。这样做的结果是,早期的研究“对大规模文本上一般应用的研究没有任何贡献”,并且“除了对语义和句法范畴有很强限制的子语言(sub-language)外,任何其他情况下都是不可用的”(Ide and Walker 1992)。

当我们研究的方向从子领域转移到大规模语料库和鲁棒系统上时,词汇覆盖问题逐渐被

重视起来,部分也是因为语音识别的研究。从语料库中获取词汇信息的最早期工作之一是 Walker and Amsler(1986)在 SRI 国际会议上展示的 FORCE4 系统。此后,词汇获取就成为了统计自然语言处理的一个最活跃的领域。

词汇获取的下一步需要做什么呢? 一个重要的倾向是更努力地寻找先验知识,这些知识可以约束词汇获取的处理过程。这是相对于早期“白手起家”,对语料中任何东西都感兴趣的研究工作而言的。先验知识可以是离散的,比如词网(WordNet)中的层次信息,也可以是概率的。例如,一个宾语名词类上的先验分布来自于动词词典,并且这个先验分布在一个语料库上被精炼。词汇获取工作在自动学习过程中将会允许简单的先验知识的方便插入和简单的错误纠正。

先验知识的一个重要资源就是语言学理论,语言学理论在统计自然语言处理中还没有被充分地利用。除了我们在这里讨论的使用语言学观点来制约词汇获取工作外,我们涉及到的读物还有 Pustejovsky et al.(1993),Boguraev and Pustejovsky(1995)以及 Boguraev(1993),它们都把语言学理论作为获取的基础。后两篇文章总结了 Cambridge University 在计算词典学上所做的重要工作,细节描述参见(Boguraev and Briscoe 1989)。这些工作虽然是非统计学的,但是包含了一个重要的想法:怎样把理论语言学和从词汇资源中获取经验结合起来。

除了语料库外,词典是词汇获取中惟一重要的资源。其他一些资源包括百科全书、类义辞典、地名词表、技术词表和其他任何参考数据,这些数据可能会提供非常用词和名字的句法和语义特征。

读者可能会奇怪,为什么我们把词汇获取限制到文本资源上? 关于语音、图像和视频的信息呢? 把词汇获取专注于文本是因为,词汇作为内容操作符比从视频和视觉数据中自动抽取出的特征含有较少的歧义。但是我们相信,随着语音识别和图像理解处理的发展,我们将有能力在非文本的多媒体数据提供的丰富的上下文中处理词汇获取的问题。研究表明,一个受过中等教育的人一年中要读到大约 100 万个词,但是他听到的词要 10 倍于此。如果我们能够通过使用更高级的信息源成功地模仿人类的语言获取,那么词汇获取的效率就会有重大的提高。

8.7 深入阅读

几本关于词汇获取的书籍和杂志特刊包括:(Zernik 1991a),(Ide and Walker 1992),(Church and Mercer 1993)和(Boguraev and Pustejovsky 1995)。很多当前的研究工作的进展可以在以下的杂志中查询:Computational Linguistics, Natural Language Engineering 和 Computers and the Humanities。在下文中,我们给读者指出了一些我们没有涉及到的词汇获取的研究工作。

另外一些解决附着歧义问题的方法是:基于转化的学习法(transformation-based learning)(Brill and Resnik 1994)和对数线性模型(loglinear models)(Franz 1997)。Collins and Brooks(1995)使用回退(back off)模型处理数据稀疏问题。罗曼斯语中也有名词短语的附着歧义问题。参见(Bourigault 1993)对法语的附着歧义和(Basili et al. 1997)对意大利语的附着歧义的论述。

Li and Abe(1995)提出了一种解决选择倾向获取问题的方法,这种方法使用了最小描述长度框架(Minimum Description Length framework),它可以代替 Resnik 的基于信息论的方法。在(Li and Abe 1996)的工作中已经考虑了一个动词的两个或多个论元之间的依赖关系。例如,drive 可以用 car 作为主语(This car drives well),但是这样它就不能有宾语出现了。只有当我们同时

考虑动词的所有论元时,才能发现这样的规则形式。感兴趣的读者也可以参阅(Velardi and Pazienza 1989)和(Webster and Marcus 1989),其中讲述了选择倾向的早期研究工作(非概率的,但是基于语料库)。

当我们得到了一个动词选择倾向的信息后,可以使用这个知识来得到子范畴框架,关于子范畴的内容在本章的第一部分已经涉及到了。Poznański and Sanfilippo(1995)和 Aone and McKee(1995)就使用了这种方法。例如,如果一个动词有 *beneficiary* 和 *recipient* 这种类型的宾语名词短语,那么它很可能就有子范畴 to-PP。

除了语义相似性外,自动的层次信息获取也是语义获取领域中的另一个重要研究方向。Hearst and Schütze(1995)和 Hearst(1992)描述的系统可以把一个新词插入已有的语义层次网络中,Coates-Stephens(1993)和 Paik et al.(1995)对专有名词也做了相同的工作。Riloff and Shepherd(1997)和 Roark and Charniak(1998)假设了一个无层次的范畴结构,为词汇指定范畴(也可以看成是一个简单的语义层次网)。

其他一些我们试图从语料库中获取的重要语义信息是反义词(Justeson and Katz 1991)和比喻(Martin 1991)。

上面指出的非文本数据是值得开发的信源。当自动构建话语上下文表达的问题已经得到解决之后,一些研究项目已经开始研究怎样利用这些数据的优点进行词汇获取。Suppes et al.(1996)强调了语言形式和上下文意思之间的行为方向(action-oriented)映射的重要性。Siskind(1996)证明了即使上下文表示有很高的歧义(在实际的学习情况中可以预期的),词汇获取也可以成功地进行。

作为获取意思的最后一个信息源,我们论及了使用词态学(morphology)来达到词汇获取的目的。例如,一个词态规则可以暗示一类特殊的意思是进行时态的。在英语中,只有非静态动词可以是进行时的。简言之,在语料库中我们发现了 *he is running*,但是没有发现 *he is knowing*,从这样的事实我们可以推断出 *know* 是静态动词,*run* 是非静态动词。参见(Dorr and Olsen 1997),(Light 1996)和(Viegas et al. 1996)所论及的上述观点。虽然这些论文都没有使用统计学方法,但是完全可以使用统计学方法来处理这样的语态信息。

最后,我们给出两个非统计学工作的重要研究领域。这两个领域有很大的潜在重要性,因为它们结合了统计方法和符号方法(如搭配的正则表达和后过滤(Justeson and Katz 1995b)),或者因为它们给出的方法除了在非统计学框架中应用外,也可以应用到统计学框架中,为进一步的统计学工作提供了大量的有用资源。

第一个领域是从电子词典中构建句法和语义知识基础的工作,这是由 Boguraev and Briscoe(1989)和 Jensen et al.(1993)提出的。这两本书非常适于那些想要了解词典在词汇获取中具有的优点和缺点的人。我们关注基于语料库的获取是因为统计自然语言处理的偏爱,但是我们相信大部分的进一步工作应该把基于语料和基于词典的获取结合起来。

第二个领域是使用与自然语言处理相关的正则表达匹配。参见(Appelt et al. 1993),(Jacquemin 1994),(Voutilainen 1995),(Sproat et al. 1996)和(Jacquemin et al. 1997),这里列出了在词汇获取中的现象和处理步骤,处理的仅仅是符号信息,并且可以根据正则语言很好地模式化。英语的符号化(tokenization of English)就是一个例子。在这种情况下,有限状态自动机(finite state automata)有其他方法无法比拟的快捷性和简洁性(Roche and Schabes 1997; Levine et al. 1992)。

8.8 习题

- 8.1 [★]结合低可靠性暗示会引发一个潜在的问题,即如果我们把所有的暗示都组合到一个正则表达式模式中,这些低可靠性暗示会“稀释”高可靠性暗示的效果,从而导致低的召回率。我们应该怎样修改假设检验来处理这个问题?提示:考虑多项式分布。
- 8.2 [★]假设某个动词有一个非常罕见的子范畴框架,讨论使用 Brent 和 Manning 的方法确定这个框架的困难度。
- 8.3 [★]如果事件空间定义为可能匹配此框架的动词出现的集合,试问这种方法会对低频的子范畴框架 f^j 的假设检验更有效吗?如果某个动词大部分是作为及物动词出现的(带有直接宾语 NP),但是有时也充当不及物动词(带有介词短语)。对于该动词,上述方法实际上削弱了它作为不及物动词出现的可能性。但是,通过适当地缩小事件空间的大小,就可以有效地避免出现上述问题,请读者讨论这种方法的优缺点。
- 8.4 [★]使用预先设定的置信度的方法(在 Brent 的方法中使用的是($\alpha = 0.02$)和绝对分类策略。动词只具有一个特殊的框架,是/不是?)存在一个难题:我们需要确定一个极限,确保尽可能多的子范畴分类是正确的(高的精确率),同时希望尽可能少的框架被遗漏(高的召回率)。在概率模型中,我们采用 $P(f^j | v^i)$ 来代替绝对的二值划分,试问这种方法是否在一定程度上缓解了上面的难题?
- 8.5 [★]在基于句法分析和先验知识的子范畴获取方法中,如何把概率句法分析和先验知识结合到一个子范畴框架的概率估计体系中?假设先验知识给定的形式是 $P(f^j | v^i)$,并且通过对一个语料库的句法分析(parsing)得出了很多概率估计 $P(s_k | f^j)$ (当动词 v^i 在句子 k 中带有框架 f^j 出现时,句子 k 的概率)。
- 8.6 [★]我们知道,如果按照通常的最大似然估计方法,数据稀疏时结果的精确率要下降。请读者借鉴第 6 章介绍的方法来修改这里的概率估计程序。Hindle and Rooth(1993)在实验中使用的是“加 1 法”。
- 8.7 [★]Hindle and Rooth(1993)采用了一个经过部分句法分析的语料库来确定 $C(v, p)$ 和 $C(n, p)$ 。请读者讨论一下,我们是否可以使用一个未经句法分析的语料库,并且这种做法可能会导致出现什么问题。
- 8.8 [★]考虑带有两个以不同介词开头的介词短语的句子,例如, *He put the book on Churchill in his backpack*。我们开发的模型将会把 *on Churchill* 附着于 *put* (使用前置词 *on*),把 *in his backpack* 附着于 *book* (使用前置词 *in*)。但这是一个不正确的分析树,因为存在交叉括号。设计一个模型来处理这样的句子(带有两个以不同前置词开头的介词短语)。
- 8.9 [★★]设计一个模型,可以解决形如 $V \cdots N \cdots PP \ PP$ 这样的句子中第二个介词短语的附着问题。我们可以看出句子中存在三种可能的附着情况:附着于动词,附着于名词或者附着于第一个介词短语中的名词。
- 8.10 [★]注意下面两种情况的不同之处:(a) 本节中附着歧义的获取方法;(b) 上一节中子

范畴框架的获取方法与第5章中搭配的获取方法。在介词短语附着关系的情况下,我们感兴趣的是什么是可以预测的。选择的模式要能最好地匹配我们能够从语料库中预测的事件(例如,一个以 *in* 开头的介词短语出现在 *send* 后面);在子范畴和搭配的情况下,我们感兴趣的是什么是不可预测的,也就是说,如果模型是正确的,一些模式就不会出现。讨论两者的区别。

8.11 [★]正如上面指出的,我们可以使用选择倾向模型计算名词的不同语义的关联强度,继而对名词进行消歧。这个方法假设我们已经知道了名词的各个语义和它所属的类别。试问如何使用选择倾向来发掘我们未知的名词语义?

8.12 [★]动词同样存在歧义问题,例如 *fire* 在下面两个句子中的情况。

a. The president fired the chief financial officer.

b. Mary fired her gun first.

(8.36)

如何使用模型对动词消歧?请读者考虑以下两种情况,一种是我们有标注动词语义的训练集,另一种没有这样的训练集。

8.13 [★]本章中讨论的模型指定的名词语义带有最大的关联强度,但这种方法并没有考虑名词的先验概率。一般说来,人们不倾向于选择稀有的名词语义,即使它对一个动词宾语的名词短语最适合。

例如,在句子 *John was reputed to be a crack shot* 中,名词 *shot* 有一个很少见的意思 *marksman*。所以,从理论上讲我们将会选择这个语义作为句子 *John fired a shot* 中的 *shot* 的语义,相应的句子的意思也变为: *John laid off a marksman*。

怎样使用先验概率来避免这样的错误?

8.14 [★★]在上面提到的方法中,认为词网(WordNet)是一个非层次的名词集,但实际上它是分层次的,那么怎样使用这些分层次的名词类信息(例如,类 *dog* 是 *animal* 的子类, *animal* 又是 *entity* 的子类)?

8.15 [★★]我们同样可以认为动词是分层次的。试问如何使用动词的层次信息得到更好的参数估计?

8.16 [★]关于本节中的模型的一个假设是,宾语名词短语的中心词决定了宾语名词短语和动词选择倾向的相容性。但是,正如 Resnik (1996: 137) 指出的,这个假设并不总是成立的。例如,在包含有否定形式 (*you can't eat stones*) 和某些形容词修饰 (*he ate a chocolate firetruck; the tractor beam pulled the ship closer*) 的句子中,无论是 *stones* 还是 *firetruck* 都不能和 *eat* 兼容,但它们仍然是定义良好的句子。请读者讨论这个问题。

8.17 [★]Hindle and Rooth (1993) 通过迭代的方法来估计模型参数,首先估计初始模型参数,对一些歧义归属问题消歧,然后在消歧实例的基础上重新估计模型参数。怎样使用这样的方法估计式 (8.34) 中名词类的先验概率?这样做的目的是改进公式中假设名词类具有的均匀分布。

8.18 [★]Resnik 的模型把关联强度作为选择倾向强度的一个比值。另一种方法是把选择倾向公式化为 $P(n|v)$,这两种方法导致了一些有趣的差别。例如,比较带有相同 $P(n|v)$ 值的两

个名词-动词对,即 $P(n_1|v_1) = P(n_2|v_2)$ 。如果 v_1 的选择倾向强度远大于 v_2 的,那么我们可以得到 $A(v_1, c(n_1)) \ll A(v_2, c(n_2))$ 。那么在这种情况下,两种方法做出了不同的预测。试讨论它们的区别。

- 8.19 [★]基于相似性度量一般化的方法依赖于假设:相似的事情也有相似的行为。如果两个词汇相似性指示一个相同的概念,那么这个假设是成立的。但是,如果词汇相似性指示不同的概念,基于相似性的一般化就会给出不同的结果。

找一些有这样的潜在危险的例子,也就是说找一些词,它们在一个方面的行为是非常相似的,而在另一个方面的行为又是非常不同的。

- 8.20 [★]基于相似性和基于类的一般化方法之间其实有很紧密的联系。基于相似性的一般化方法考虑了最近邻的那些词,然后通过和这些词之间的相似度的加权和得出最后的结果。基于类的一般化则选取了一个最有可能和它相似的类别。在最简单的情况下,例如新词,一般取类中所有元素的平均值作为结果。其实两种方法可以互相转化,因为在基于相似性的方法中,每个近邻对象都可以被当成一个单独的类,同样,在基于类的一般化方法中,我们采用的方法可以是收集所有的类别信息,即把词和每个类之间的近似程度有机地结合起来求加权和,那么在一定程度上,它可以被看成是基于相似性的一般化方法。

请读者讨论两种一般化方法之间的联系,并说明效率因素在其中充当什么样的角色?

- 8.21 [★]如图 8.3 所示的同现矩阵表示了不同类型的信息,这些信息依赖于不同的同现定义。对于下面的同现定义,你认为 *fire* 的相似词将会是什么词汇类型:文本中的同现;句子中的同现;在一个词右边三个词之内的同现;一个词右边紧邻的同现。Finch and Chater(1994)和 Schütze(1995)的研究表明最后一种紧邻同现可以用来发现句法结构。

- 8.22 [★★]我们已经考虑过的方法能够比较向量和概率分布这样的简单对象。但是,我们还希望衡量更复杂的对象(如 *tree*)中的语义相似性,参见(Sheridan and Smeaton 1992)。怎样衡量 *tree* 之间的语义相似性?这样的方法怎样才能使词汇之间的语义相似性度量比使用“臃肿”结构的度量方法效果更好?

- 8.23 [★]选择图 8.3 的列中两个开头的词,对图 8.3 至图 8.5 的三个矩阵中的每一个,使用表 8.7 中的每一种方法分别计算相似性。

- 8.24 [★]证明如果两个向量有相同数量的非零元素,那么 Dice 系数和余弦系数的值是相同的。

- 8.25 [★]语义相似性可能是上下文相关的。例如电子和网球,当我们讨论它们的外形(两者都是球形的)时就是相似的,当我们讨论它们的大小时就是不相似的。

讨论相似性在多大程度上是上下文相关的,并且在什么时候它会影响到正确性的一般化。

- 8.26 [★]证明平均值偏差(divergence to the average)(IRad)的上界是 $2\log 2$ 。

- 8.27 [★]选择图 8.3 的列中两个开头的词,对图 8.3 至图 8.5 的三个矩阵中的每一个,使用表 8.9 中的每种方法分别计算非相似性。计算相对熵(KL divergence)时要对概率进行平滑。对于相对熵,非相似性度量是非对称的吗?

8.28 [★★]实际上, L_1 范式和欧氏范式都是 Minkowski 范式 L_p 的特殊形式:

$$L_p(a, b) = \sqrt[p]{\sum_i |a_i - b_i|^p} \quad (8.49)$$

欧氏范式又称为 L_2 范式。 L_1 范式要比欧氏范式更适合于处理概率分布。

另一个用来处理向量的范式 L_∞ , 也就是 $p \rightarrow \infty$ 的 L_p (Salton et al. 1983)。 L_∞ 相应的方程是什么?

8.29 [★]当表 8.9 中的任意一种度量方法得出 0 的非相似性时, 也意味着其他两种方法也会得出 0 的非相似性吗? 请读者说明。

8.30 [★]如果相应于表 8.9 中的一种方法, 两个概率分布有最大的非相似性, 例如 IRad $(p, q) = 2\log 2$, 那么这也意味着对于其他两种方法也有最大的非相似性吗?

第三部分 语 法

第 9 章 马尔可夫模型

第 10 章 词性标注

第 11 章 概率上下文无关文法

第 12 章 概率句法分析

第9章 马尔可夫模型

隐马尔可夫模型(Hidden Markov Model, 简称 HMM)已经成为现代语音识别系统中构建统计模型的重要手段。尽管它存在很多不足之处,但是各种 HMM 的变形仍然被广泛应用于这个领域中,而且它们一般也被认为是最成功的技术。在这一章中,我们将展开介绍 HMM 的基本思想,阐述它们的具体应用,最后总结一下 HMM 模型的扩展以及它在实际工程中的具体应用。

从某种意义上讲,HMM 本身是一个马尔可夫过程的概率函数。在第2章和第6章中的 n 元语法模型中,我们已经介绍了一个马尔可夫过程的例子。马尔可夫过程(链/模型)最早由 Andrei A. Markov (Chebyshev 的一个学生)提出¹。它的最原始目的也是为了语言上的应用,即为俄国文学作品中的字母序列建模(Markov 1913),但是马尔可夫模型随之发展成为了一个通用的统计工具。为了区别于传统的 HMM,我们把一般马尔可夫模型称为显马尔可夫模型(Visible Markov Model, VMM)。

本章是本书语法部分的起始章节,因为句子中的词语序列的问题是理解语法结构的基本前提。接下来我们将会看到的 VMM 便是这样的一个明证。HMM 通过假定额外的隐含结构,使模型在一个更高的抽象水平上进行操作,并且允许我们查看词语类别顺序。本章首先提出 HMM 的基本思想,然后介绍 HMM 在标注方面的应用。这个部分的最后两章将阐述核心语法(类似短语结构语法)的概率形式化问题。

9.1 马尔可夫模型

我们经常会考虑一个由并不互相独立的随机变量组成的序列(或许是根据时间的前后),序列中每个变量的值依赖于它前面的元素。对于很多类似的系统,假设所要预测的将来的随机变量值的依据就是现在的随机变量的值,从直观上来说这是合理的,但是我们并不需要了解序列中所有过去的随机变量的值。例如,如果某个随机变量表示大学图书馆中书的数目,那么知道今天图书馆中有多少本书足以预测明天会有多少本书,其中并不需要额外知道上周图书馆里有多少本书,也不必考虑上一年的情况。也就是说,序列中未来的元素在给定当前元素的情况下与过去的元素是条件独立的。

假设 $X = (X_1, \dots, X_T)$ 是一个取值于有限集合 $S = \{s_1, \dots, s_N\}$ 的随机变量序列, S 称为状态空间,马尔可夫模型的性质如下:

有限视野:

$$P(X_{t+1} = s_k | X_1, \dots, X_t) = P(X_{t+1} = s_k | X_t) \quad (9.1)$$

时间不变性(固定的):

$$= P(X_2 = s_k | X_1) \quad (9.2)$$

X 称为一个马尔可夫链,或称具有马尔可夫性质。随机转移矩阵 A 可以描述该马尔可夫链:

$$a_{ij} = P(X_{t+1} = s_j | X_t = s_i) \quad (9.3)$$

其中, $a_{ij} \geq 0, \forall i, j$ 并且 $\sum_{j=1}^N a_{ij} = 1, \forall i$ 。

另外我们需要指定 π , 它表示马尔可夫链中不同初始状态的概率:

$$\pi_i = P(X_1 = s_i) \quad (9.4)$$

其中, $\sum_{i=1}^N \pi_i = 1$ 。为了避免出现这个向量,我们指定马尔可夫模型总是以某个特定的额外状态 s_0 作为起始状态,然后通过使用包含在矩阵 A 中的状态转移来指定通常记录在 π 中的概率。

从这个一般性描述中,很明显我们在第6章中看到的 n 元语法模型就是马尔可夫模型。马尔可夫模型可以用于任何需要为事件的线性序列概率进行建模的情况下。例如,它们也可以为语音识别系统建立有效的语音序列模型,描述对话系统中的语言行为序列。

此外,我们可以用图9.1所示的状态图来表示马尔可夫链。这里,状态用包含状态名的圆圈表示,单个初始状态有一个指向它的箭头。概率转移用连接状态的箭头表示,这些弧上标有处于箭头尾部状态时发生状态转移的概率。图中转移概率为零的弧被省略了。我们需要注意的是每个状态发出的弧的概率和为1。从上述表述我们可以很明显地看出,马尔可夫模型相当于一个弧上标明概率的(不确定的)有限状态自动机。马尔可夫模型的性质保证了存在一个有限状态自动机,但是这里并没有远距离的依存关系的概念,下一步的状态仅仅取决于现在所处的状态。

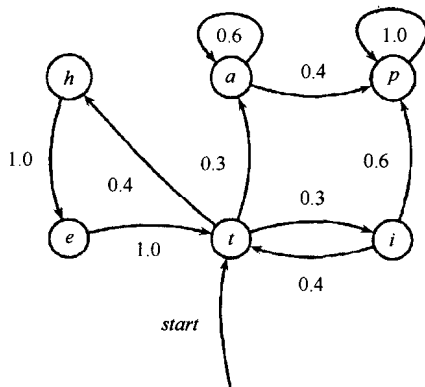


图9.1 一个马尔可夫模型的例子

在显马尔可夫模型中,机器的当前状态是已知的,因此状态序列或者它的一些确定函数被认为是输出。

马尔可夫链中的状态序列(即随机变量序列)的概率 $X_1, \dots, X_T$ 很容易计算,只需要计算弧上或者随机矩阵中的概率的乘积:

$$\begin{aligned} P(X_1, \dots, X_T) &= P(X_1)P(X_2|X_1)P(X_3|X_1, X_2) \cdots P(X_T|X_1, \dots, X_{T-1}) \\ &= P(X_1)P(X_2|X_1)P(X_3|X_2) \cdots P(X_T|X_{T-1}) \\ &= \pi_{X_1} \prod_{t=1}^{T-1} a_{X_t X_{t+1}} \end{aligned}$$

因此,使用图9.1中的马尔可夫模型,我们有:

$$\begin{aligned}
 P(t, i, p) &= P(X_1 = t)P(X_2 = i|X_1 = t)P(X_3 = p|X_2 = i) \\
 &= 1.0 \times 0.3 \times 0.6 \\
 &= 0.18
 \end{aligned}$$

在实际应用中,最重要的一点在于能否把一个过程转化成一个马尔可夫过程,因为如果不经映射通常不能直接应用。以我们在第6章所看到的 n 元语法模型为例。我们也许会想,在 $n \geq 3$ 时,这样的模型并不是马尔可夫模型,因为它违背了有限视野条件——我们查看了更早的历史信息。但是我们可以简单地通过把适量的历史信息转化成状态空间(状态是 $(n-1)$ 元的,例如(*was, walking, down*)就是四元语法模型中的一个状态)来把 n 元语法模型重构成显马尔可夫模型。一般来说,有限的历史信息通常都可以通过这种方式简单地把状态空间作为前面多个状态的向量积(crossproduct)来转换。在这些例子中,我们有时会说一个 m 阶的马尔可夫模型, m 是我们用来预测下个状态所用到的前面的状态数。因此,注意一个 n 元语法模型等同于一个 $(n-1)$ 元马尔可夫模型。

9.2 隐马尔可夫模型

在隐马尔可夫模型(HMM)中,模型所经过的状态序列是未知的,我们只知道它的一些概率函数。

例 1 假设你有一台软饮料机,它可以处于两种状态,可乐(CP)或者冰茶(IP),但是它在每一次售出之后的状态都是随机切换的,如图 9.2 所示。

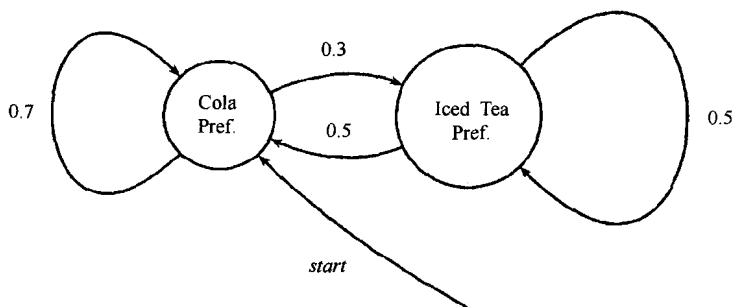


图 9.2 表示了机器的状态和状态转移概率的软饮料机

现在,当你放进硬币以后,如果机器处于可乐状态时它出来的就是可乐,而处于冰茶状态时出来的就是冰茶,于是我们便有了一个显马尔可夫模型。但如果不是这样,它只是趋向于这么做。因此,我们需要符号化观测到的发射概率:

$$P(O_t = k | X_t = s_i, X_{t+1} = s_j) = b_{ijk}$$

对于这台机器,输出实际上是与 s_j 独立的,因此可以用下面的概率矩阵来描述:

	从状态到输出概率		
	cola	iced tea (ice_t)	lemonade (lem)
CP	0.6	0.1	0.3
IP	0.1	0.7	0.2

(9.5)

如果机器总是从可乐状态开始,那么看到的输出序列是 $\{lem, ice_t\}$ 的概率是多少?

解答:

我们需要考虑 HMM 所要经过的所有路径,然后对它们进行求和。我们知道机器从 CP 开始有依赖于在其他两个时间内机器所处状态的四个概率,因此总的概率是:

$$\begin{aligned} & 0.7 \times 0.3 \times 0.7 \times 0.1 + 0.7 \times 0.3 \times 0.3 \times 0.1 + \\ & 0.3 \times 0.3 \times 0.5 \times 0.7 + 0.3 \times 0.3 \times 0.5 \times 0.7 = 0.084 \end{aligned}$$

9.2.1 为什么使用隐马尔可夫模型

当系统中表层事件可能是由底层事件引发的时,HMM 能有效地解决这类问题。一个广泛的应用是标注系统——为文本中的每个词分配一个词性标注(或者是其他的类别)。我们可以认为文本中实际的词是由底层的马尔可夫词性标注链生成的。我们将在第 10 章中讨论这个模型。

如果这个模型合适,那么很重要的一点是:它是一类可以通过使用 EM(Expectation Maximization)算法进行有效训练的模型。给出假设由 HMM 生成的足够的数据(这些 HMM 的模型结构是固定的,但是弧上的概率却是变化的),这个算法允许我们通过自动学习得到描述观测数据的最佳模型参数。

另一个说明如何使用 HMM 的简单例子是生成 n 元语法模型的线性插值参数。我们在第 6 章中讨论了一种方法来估计句子概率:

$P(\text{Sue drank her beer before the meal arrived})$

可以使用 n 元语法模型,例如三元语法模型,但是如果仅仅使用固定 n 的 n 元语法模型,会受到数据稀疏的影响。回忆 6.3.1 节中平滑 n 元估计的一种方法,即对于不同的 n 使用 n 元估计的线性插值,例如:

$$P_{\text{H}}(w_n | w_{n-1}, w_{n-2}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n | w_{n-1}) + \lambda_3 P_3(w_n | w_{n-1}, w_{n-2})$$

这样我们可以了解一个特殊词出现的可能性,即使三元语法模型的覆盖率很稀疏。于是这个问题变成了怎样设置参数 λ_i 。尽管我们能够对可以使用哪些参数值做出合理的猜测(它们同时都必须遵守随机条件限制 $\sum_i \lambda_i = 1$),但应该能自动找出最优值,而实际上我们也可以做到这一点(Jelinek 1990)。

关键点在于可以使用隐藏状态构建一个 HMM,而隐藏状态表明了我们是选择使用一元、二元还是三元概率。HMM 的训练算法会确定进入这些隐藏状态的弧的权重的最优值,这些权重值又代表了每个 n 元语法模型通过设定上面的参数 λ_i 来决定的概率值。

具体来说,我们为每个词语对构建一个四个状态的 HMM,其中一个代表基本的词语对,另外三个代表用来计算下一个转移的 n 元语法模型的选择。图 9.3 给出了 HMM 的一部分。注意 HMM 是怎样像前面的方程一样分配概率的: w^c 有三种可能跟随 $w^a w^b$,看到下一个是 w^c 的总的概率等于每条弧上的 n 元概率乘以对应的参数 λ_i 的和。本章提出的 HMM 训练算法也可以应用于这个网络,用来改善参数的初始估计。但是有两点值得注意,(1) 这种转化是通过加入 epsilon 转移实现的,这是一种并不产生输出符号的转换;(2) 正如前面所列出的,每个词语对都有独立的参数 λ_{iwb} 。但是,我们并不希望独立地调节这些参数,因为这将使数据更加稀疏。而且,对于一个固定的 i ,我们希望让所有(至少是多个类)的参数 λ_{iwb} 具有同样的值,为了实现这一点我们使用

了限制型状态。如何扩张这些基本的 HMM 模型的问题将在 9.4 节中讨论。

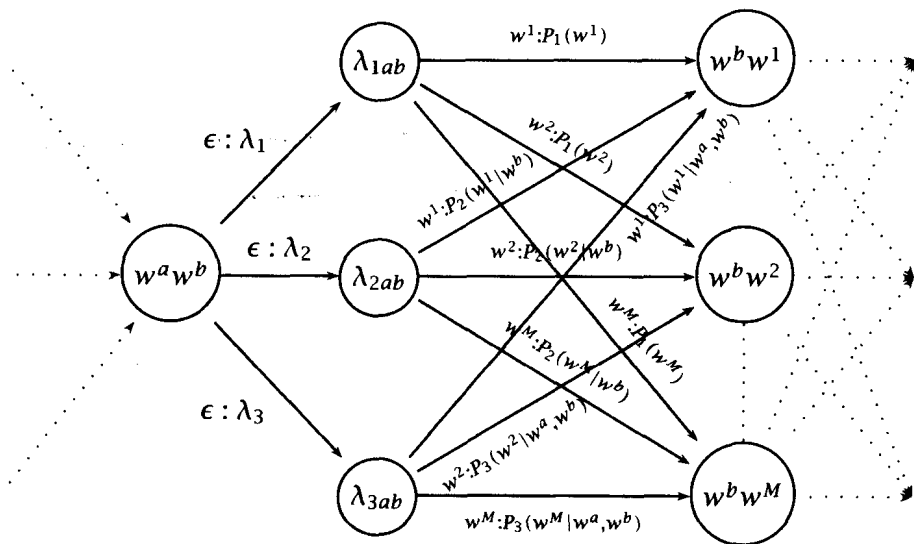


图 9.3 线性插值 HMM 语言模型的一部分。弧上的符号 $o:p$ 表示转移概率是 p , o 是转移完成后的输出 (概率是 1)

9.2.2 隐马尔可夫模型的一般形式

一个 HMM 可以由一个五元组 (S, K, Π, A, B) 表示, 其中, S 和 K 分别是状态集合和输出字母表, Π , A 和 B 分别是初始状态、状态转移以及符号发射概率。我们在这一章中所用到的符号在表 9.1 中做了小结。随机变量 X_t 从状态名映射到对应的整数。在这里展示的版本中, t 时刻发射的符号依赖于 t 和 $t+1$ 时刻的状态。这有时候也被称为弧发射 HMM, 因为我们可以把符号看成是来自弧, 正如图 9.3 所示。另一种形式是状态发射 HMM, 即 t 时刻发射的符号只依赖于 t 时刻的状态。例 1 中的 HMM 便是一个状态 HMM。但是我们可以把它看成是一个弧发射 HMM, 通过简单地建立参数 b_{ijk} , 使得 $\forall j', j'', b_{ij'k} = b_{ij''k}$ 。在 9.4 节中我们将对此进行详细讨论。

表 9.1 本章的 HMM 中使用的符号

状态集	$S = \{s_1, \dots, s_N\}$
输出字母表	$K = \{k_1, \dots, k_M\} = \{1, \dots, M\}$
初始状态概率	$\Pi = \{\pi_i\}, i \in S$
状态转移概率	$A = \{a_{ij}\}, i, j \in S$
符号发射概率	$B = \{b_{ijk}\}, i, j \in S, k \in K$
状态序列	$X = (X_1, \dots, X_{T+1}) \quad X_t: S \rightarrow \{1, \dots, N\}$
输出序列	$O = (o_1, \dots, o_T) \quad o_t \in K$

下面给出一个 HMM 的详细说明, 我们可以看到它直接模拟了一个马尔可夫过程的运行情况, 并且产生了输出序列。图 9.4 中的程序实现了这个过程。但是, 就它本身而言这么做并不是很有意思。HMM 的优势在于它假设部分数据集是由 HMM 生成的, 因而我们能够计算概率和可能的底层状态序列。

```

1  t := 1;
2  Start in state  $s_i$  with probability  $\pi_i$  (i.e.,  $X_1 = i$ )
3  forever do
4      Move from state  $s_i$  to state  $s_j$  with probability  $a_{ij}$  (i.e.,  $X_{t+1} = j$ )
5      Emit observation symbol  $o_t = k$  with probability  $b_{ijk}$ 
6      t := t + 1
7  end

```

图 9.4 描述马尔可夫过程的一段程序

9.3 隐马尔可夫模型的三个基本问题

在隐马尔可夫模型中我们需要了解三个基本问题：

1. 给出一个模型 $\mu = (A, B, \pi)$, 怎样有效地计算某个观测序列发生的概率, 即 $P(O|\mu)$?
2. 给出观测序列 O 和模型 μ , 我们怎样选择一个状态序列 $(X_1, \dots, X_{T+1})$, 以便能够最好地解释观测序列?
3. 给定观测序列 O , 以及通过改变模型 $\mu = (A, B, \pi)$ 的参数而得到的模型空间, 我们怎样才能找到一个最好地解释这个观测序列的模型?

一般来说, 我们处理的问题并不像软饮料机问题那么简单。模型的参数无从知晓, 所以我们不得不从数据中估计, 这是上面的第三个问题。第一个问题可以用来确定最佳模型。第二个问题让我们猜测出马尔可夫链中哪条路径才是最有可能经过的, 而且这条隐藏路径能够被用来解决分类问题, 例如我们将会在第 10 章中看到它在标注方面的应用。

9.3.1 计算观测序列的概率

给定观测序列 $O(o_1, \dots, o_T)$ 和模型 $\mu = (A, B, \pi)$, 我们想知道怎样有效计算 $P(O|\mu)$, 即给定模型下观测序列的概率。

对于任意状态序列 $X = (X_1, \dots, X_{T+1})$,

$$\begin{aligned}
 P(O|X, \mu) &= \prod_{t=1}^T P(o_t|X_t, X_{t+1}, \mu) \\
 &= b_{X_1 X_2 o_1} b_{X_2 X_3 o_2} \cdots b_{X_T X_{T+1} o_T}
 \end{aligned} \tag{9.6}$$

而且,

$$P(X|\mu) = \pi_{X_1} a_{X_1 X_2} a_{X_2 X_3} \cdots a_{X_T X_{T+1}} \tag{9.7}$$

现在,

$$P(O, X|\mu) = P(O|X, \mu)P(X|\mu) \tag{9.8}$$

因此

$$\begin{aligned}
 P(O|\mu) &= \sum_X P(O|X, \mu)P(X|\mu) \\
 &= \sum_{X_1 \cdots X_{T+1}} \pi_{X_1} \prod_{t=1}^T a_{X_t X_{t+1}} b_{X_t X_{t+1} o_t}
 \end{aligned} \tag{9.9}$$

这个推导是很直接的。在例 1 中我们用这种方法计算出了观测序列的概率。可以简单地 对可能的状态序列发生的观测概率求和。但遗憾的是,直接评估表明表达式效率很低。对于 一般的情况(可以从任意状态开始,每步转移到另外任何一个状态),这种直接计算的方法需要 $(2T+1) \cdot N^{T+1}$ 次乘法。

避免复杂度的秘诀是利用动态编程和记忆,即记住部分结果而不是每次都重新计算。这 个一般的概念在计算语言学中的其他很多地方都得到了应用,比如 chart 分析,而且在计算机 科学中更为常见,参见 (Cormen et al.1990:ch.16) 中的一般性介绍。对于类似 HMM 之类的算 法,动态编程问题通常用格子来描述(或称之为格路)。这里,我们构造一个状态相对于时间的 方阵,根据前面时间所处状态的概率来计算处于现在每个状态的概率。图 9.5 和图 9.6 做了 很好的阐述。一个格可以记录在特定时间以特定状态结尾的 HMM 的所有初始子路径的概率, 这样就可以根据较短子路径计算得出较长子路径的概率。

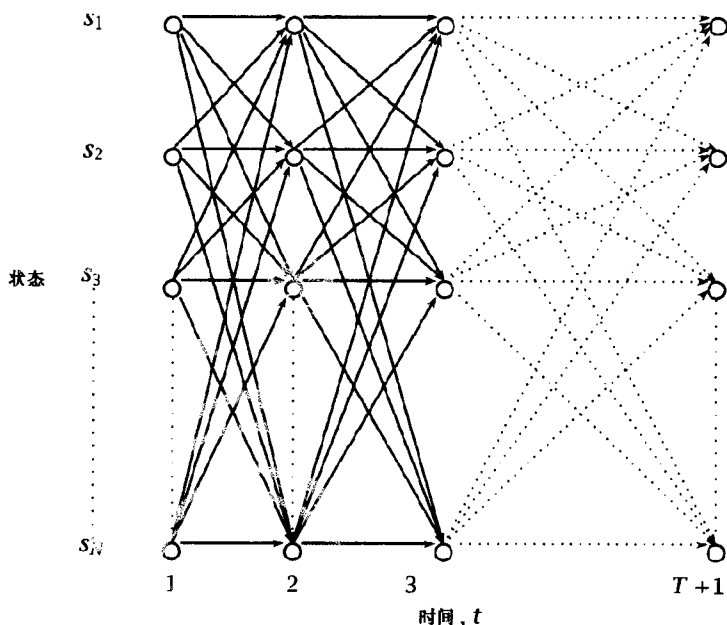


图 9.5 格路算法。格路由状态相对于时间组成的一个方阵。一个节点 (s_i, t) 可以存储包含 $X_t = i$ 的状态序列信息。连线表明了节点之间的联系。这里我们得到了一个全连接的 HMM, 在每一步都可以从任意状态转到其他状态

前向过程

图中已经指明的存储形式称为前向过程,我们用前向变量来描述它。

$$\alpha_i(t) = P(o_1 o_2 \cdots o_{t-1}, X_t = i | \mu) \quad (9.10)$$

前向变量 $\alpha_i(t)$ 存储在格路的 (s_i, t) 中,表示在 t 时刻以状态 s_i 结束时总的概率(给定观测序列 $o_1 \cdots o_{t-1}$),可通过对格路节点的所有入弧的概率进行求和计算得到。我们使用如下过程从左到右计算格路中的前向变量:

1. 初始化

$$\alpha_i(1) = \pi_i, \quad 1 \leq i \leq N$$

2. 推导

$$\alpha_j(t+1) = \sum_{i=1}^N \alpha_i(t) a_{ij} b_{ij|o_t}, \quad 1 \leq t \leq T, 1 \leq j \leq N$$

3. 求和

$$P(O|\mu) = \sum_{i=1}^N \alpha_i(T+1)$$

这是一个非常廉价的算法,它只需要 $2N^2 T$ 次乘法。

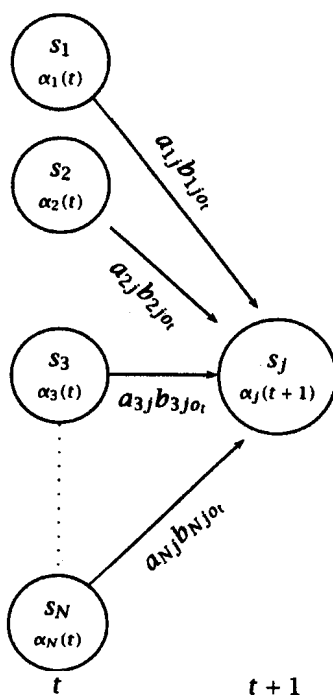


图 9.6 格路算法:在某个节点的前向概率计算的描述。前向概率 $\alpha_j(t+1)$ 可以通过对每个入弧的概率与起始节点的前向概率的乘积进行求和计算得到

后向过程

很明显我们并不一定需要像这样一直存储前向计算的结果,我们也可以进行后向计算。后向过程计算了后向变量,后向变量中记录了在给定当前状态 S_i 和 t 时刻的情况下观测序列中剩余部分的概率之和。结合使用前向概率和后向概率,对于解决第三个问题中的参数重估很关键,这才是我们引入它的真正原因。

定义后向变量

$$\beta_i(t) = P(o_t \cdots o_T | X_t = i, \mu) \quad (9.11)$$

于是我们可以通过如下方法从右至左计算格路中的后向变量:

1. 初始化

$$\beta_i(T+1) = 1, \quad 1 \leq i \leq N$$

2. 推导

$$\beta_i(t) = \sum_{j=1}^N a_{ij} b_{ij} o_t \beta_j(t+1), \quad 1 \leq t \leq T, 1 \leq i \leq N$$

3. 求和

$$P(O|\mu) = \sum_{i=1}^N \pi_i \beta_i(1)$$

表 9.2 表明了例 1 中在给定观测序列为 $O(\text{lem}, \text{ice_t}, \text{cola})$ 时软饮料机的前向变量和后向变量以及我们后面会遇到的其他变量的计算。

表 9.2 对于 $O = (\text{lem}, \text{ice_t}, \text{cola})$ 的变量计算

时间(t):	输出			
	lem	ice_t	cola	
	1	2	3	4
$\alpha_{CP}(t)$	1.0	0.21	0.0462	0.021 294
$\alpha_{IP}(t)$	0.0	0.09	0.0378	0.010 206
$P(o_1 \cdots o_{t-1})$	1.0	0.3	0.084	0.0315
$\beta_{CP}(t)$	0.0315	0.045	0.6	1.0
$\beta_{IP}(t)$	0.029	0.245	0.1	1.0
$P(o_1 \cdots o_T)$	0.0315			
$\gamma_{CP}(t)$	1.0	0.3	0.88	0.676
$\gamma_{IP}(t)$	0.0	0.7	0.12	0.324
$\hat{X}_t$	CP	IP	CP	CP
$\delta_{CP}(t)$	1.0	0.21	0.0315	0.013 23
$\delta_{IP}(t)$	0.0	0.09	0.0315	0.005 67
$\psi_{CP}(t)$		CP	IP	CP
$\psi_{IP}(t)$		CP	IP	CP
$\hat{X}_t$	CP	IP	CP	CP
$P(\hat{X})$	0.019 404			

二者结合

更一般地,实际上我们可以结合使用前向和后向存储来计算一个观测序列的概率。观察到:

$$\begin{aligned}
 P(O, X_t = i | \mu) &= P(o_1 \cdots o_T, X_t = i | \mu) \\
 &= P(o_1 \cdots o_{t-1}, X_t = i, o_t \cdots o_T | \mu) \\
 &= P(o_1 \cdots o_{t-1}, X_t = i | \mu) \\
 &\quad \times P(o_t \cdots o_T | o_1 \cdots o_{t-1}, X_t = i, \mu) \\
 &= P(o_1 \cdots o_{t-1}, X_t = i | \mu) P(o_t \cdots o_T | X_t = i, \mu) \\
 &= \alpha_i(t) \beta_i(t)
 \end{aligned}$$

因此

$$P(O|\mu) = \sum_{i=1}^N \alpha_i(t) \beta_i(t), \quad 1 \leq t \leq T+1 \quad (9.12)$$

前面的等式是这个等式的特例。

9.3.2 确定最佳状态序列

这个过程通常被称为译码。这个问题被有点模糊地描述为“找到能够最好地解释观测值的状态序列”。这是因为不只一种方法可以这样做。一种方法是独立地选择状态。即对于每个 $t, 1 \leq t \leq T+1$, 我们可以找到 X_t 使得 $P(X_t|O, \mu)$ 最大。

令

$$\begin{aligned} \gamma_i(t) &= P(X_t = i|O, \mu) \\ &= \frac{P(X_t = i, O|\mu)}{P(O|\mu)} \\ &= \frac{\alpha_i(t) \beta_i(t)}{\sum_{j=1}^N \alpha_j(t) \beta_j(t)} \end{aligned} \quad (9.13)$$

单独的最可能的状态 $\hat{X}_t$ 是:

$$\hat{X}_t = \underset{1 \leq i \leq N}{\operatorname{argmax}} \gamma_i(t), \quad 1 \leq t \leq T+1 \quad (9.14)$$

它最大化了将被正确猜测的状态的期望数目。但是,也许会得到一个可能性很小的状态序列,因此这并不是常用的方法。最常用的方法是 Viterbi 算法,它能够有效地计算最可能的状态序列。

Viterbi 算法

通常我们希望找到最可能的完整路径,即:

$$\underset{X}{\operatorname{argmax}} P(X|O, \mu)$$

要实现这一点,可以对于某个固定的 O 求最大值:

$$\underset{X}{\operatorname{argmax}} P(X, O|\mu)$$

一个有效地计算这条路径的格路算法是 Viterbi 算法。定义为:

$$\delta_j(t) = \max_{X_1 \cdots X_{t-1}} P(X_1 \cdots X_{t-1}, o_1 \cdots o_{t-1}, X_t = j|\mu)$$

格路中每个节点的变量存储了到这个节点的最可能路 Q 径的概率。对应的变量 $\phi_j(t)$ 记录了导致这条最可能路径的入弧节点。使用动态编程,我们可以采用如下方法计算通过整个格路的最可能路径:

1. 初始化

$$\delta_j(1) = \pi_j, \quad 1 \leq j \leq N$$

2. 推导

$$\delta_j(t+1) = \max_{1 \leq i \leq N} \delta_i(t) a_{ij} b_{ij} o_t, \quad 1 \leq j \leq N$$

存储回溯路径

$$\psi_j(t+1) = \arg \max_{1 \leq i \leq N} \delta_i(t) a_{ij} b_{ij|o_t}, 1 \leq j \leq N$$

3. 终止以及路径读出(通过回溯)。最可能的状态序列通过正确的回溯得到:

$$\begin{aligned} \hat{X}_{T+1} &= \arg \max_{1 \leq i \leq N} \delta_i(T+1) \\ \hat{X}_t &= \psi_{\hat{X}_{t+1}}(t+1) \\ P(\hat{X}) &= \max_{1 \leq i \leq N} \delta_i(T+1) \end{aligned}$$

在这些计算中存在一些约束。我们假设路径是随机选择的。在实际应用中,人们通常不仅仅只需要算出最佳状态序列,还需要得到可能的路径中的 n 个最佳(n -best)序列或者图形。为了实现这一点,我们通常存储一个节点的 $m < n$ 个最优的先前状态。

上面的表 9.2 表明在这两种解释下最可能状态和最可能状态序列的计算。对于这个例子而言,它们被证明是一致的。

9.3.3 隐马尔可夫模型的参数估计问题

给定一个特定的观测序列,我们希望确定模型 $\mu = (A, B, \pi)$ 的参数值,并且要求这个模型能够给出观测序列的最佳解释。使用最大似然估计(Maximum Likelihood Estimation),意味着我们需要找到能够使得 $P(O|\mu)$ 最大的参数值:

$$\arg \max_{\mu} P(O_{\text{training}}|\mu) \quad (9.15)$$

目前并没有已知的解析方法来选择 μ , 使得 $P(O|\mu)$ 最大。但是我们可以通过迭代爬山算法使得它局部最大化。这种算法被称为 Baum-Welch 或前向后向(Forward-Backward)算法,它们实际上是我们将会在 14.2.2 节中讲到的 EM(Expectation Maximization)算法的一个特例。它的工作方式如下:最初我们并不知道模型的信息,但是可以使用某个(也许是随机选取的)模型算出观测序列的概率;然后查看计算过程,即可发现哪个状态转移或者符号发射可能出现的次数最多。通过增加它们的概率,就可以选择一个修改后的模型,使得它可以为观测序列给出更高的概率。这个最大化过程通常称为训练模型,并且是在训练数据上执行的。

定义 $p_t(i, j)$ 如下, $1 \leq t \leq T, 1 \leq i, j \leq N$ 。这是在给定观测序列 O 的情况下,在 t 时刻经过某条弧的概率,参见图 9.7。

$$\begin{aligned} p_t(i, j) &= P(X_t = i, X_{t+1} = j | O, \mu) \\ &= \frac{P(X_t = i, X_{t+1} = j, O | \mu)}{P(O | \mu)} \\ &= \frac{\alpha_i(t) a_{ij} b_{ij|o_t} \beta_j(t+1)}{\sum_{m=1}^N \alpha_m(t) \beta_m(t)} \\ &= \frac{\alpha_i(t) a_{ij} b_{ij|o_t} \beta_j(t+1)}{\sum_{m=1}^N \sum_{n=1}^N \alpha_m(t) a_{mn} b_{mn|o_t} \beta_n(t+1)} \end{aligned} \quad (9.16)$$

注意, $\gamma_i(t) = \sum_{j=1}^N p_t(i, j)$ 。现在, 如果我们根据时间求和, 便可以给出期望值(计数):

$\sum_{t=1}^T \gamma_i(t)$ = 在 O 中从状态 i 发出的转移的期望数目

$\sum_{t=1}^T p_t(i, j)$ = 在 O 中从状态 i 到 j 的转移的期望数目

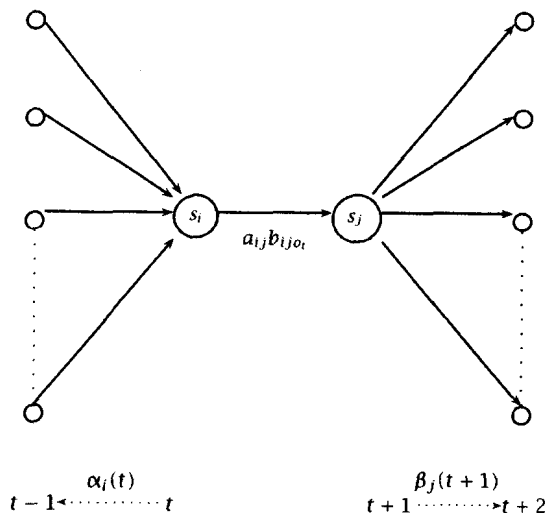


图 9.7 经过某条弧的概率。给定一个模型和一个观测序列, 我们可以计算在 t 时刻马尔可夫过程从状态 s_i 到 s_j 的概率

因此我们可以从模型 μ 开始(可能是预先选择的,也可能是随机选择的),通过现有的模型运行 O 来估计每个模型参数值的期望,接下来改变模型来最大化使用较多的路径的值(仍然要遵从随机限制)。重复这个过程直至模型收敛到参数 μ 的最优值。

重估公式如下:

$$\begin{aligned} \hat{\pi}_i &= \text{expected frequency in state } i \text{ at time } t = 1 \\ &= \gamma_i(1) \end{aligned} \quad (9.17)$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\text{expected number of transitions from state } i \text{ to } j}{\text{expected number of transitions from state } i} \\ &= \frac{\sum_{t=1}^T p_t(i, j)}{\sum_{t=1}^T \gamma_i(t)} \end{aligned} \quad (9.18)$$

$$\begin{aligned} \hat{b}_{ijk} &= \frac{\text{expected number of transitions from } i \text{ to } j \text{ with } k \text{ observed}}{\text{expected number of transitions from } i \text{ to } j} \\ &= \frac{\sum_{\{t: o_t=k, 1 \leq t \leq T\}} p_t(i, j)}{\sum_{t=1}^T p_t(i, j)} \end{aligned} \quad (9.19)$$

因此, 从 $\mu = (A, B, \pi)$ 可以派生出 $\hat{\mu} = (\hat{A}, \hat{B}, \pi)$ 。更进一步, 如 Baum 所证明的, 我们有:

$$P(O|\hat{\mu}) \geq P(O|\mu)$$

这是 EM 算法(参见 14.2.2 节)的一个一般属性。因此, 通过大量参数重估迭代可以改善我们的模型。通常, 重估参数一直进行到结果没有显著的提高为止。但是, 这个参数重估过程并不

能保证得到最好的模型,因为重估过程会停止在局部极值点上(甚至可能只是一个鞍点)。在很多颇有意思的问题中,似然函数是一个复杂的非线性表面,并且有很多局部极值点。不过, Baum-Welch 重估对于 HMM 通常是很有效的。

作为本节的结束,我们来考虑用 Baum-Welch 算法重估软饮料机 HMM 模型中的参数。如果把当前使用的模型作为初始模型,那么通过在观测序列(lem, ice_t, cola)上的训练可获得下列 $p_i(i, j)$ 值:

时间 (和j)											(9.20)
1			2			3					
	CP	IP	y ₁	CP	IP	y ₂	CP	IP	y ₃		
i	CP	0.3	0.7	1.0	0.28	0.02	0.3	0.616	0.264	0.88	
	IP	0.0	0.0	0.0	0.6	0.1	0.7	0.06	0.06	0.12	

因此参数可以进行如下重估:

		初始			重估		
Π	CP	1.0			1.0		
	IP	0.0			0.0		
A	CP	0.7	0.3		0.5486	0.4514	
	IP	0.5	0.5		0.8049	0.1951	
B	CP	cola	ice_t	lem	cola	ice_t	lem
	IP	0.6	0.1	0.3	0.4037	0.1376	0.4587
		0.1	0.7	0.2	0.1463	0.8537	0.0

9.4 隐马尔可夫模型:实现、性质和变形

9.4.1 实现

除了以上讨论的一些理论,还有很多实现 HMM 的实际问题。为了使得 HMM 标注更有效和准确,需要注意很多地方。最明显的问题是我们所计算的概率是由很小的数字相乘得到的。这种计算会很快超过计算机所能表示的浮点数的下界(即使把它存储为双精度浮点数也是如此)。

Viterbi 算法只涉及到乘积,而且选择的是最大的元素,因此我们可以采用对数来运行整个过程。这不仅解决了浮点数下溢问题,而且还加快了计算速度,因为加法比乘法快得多。在实践中,一个快速的 Viterbi 算法是非常重要的,因为这是一个实时算法,然而训练通常可以按离线方式慢慢完成。

但是,在前向-后向算法中也需要处理类似的浮点数下溢问题。它们需要执行加法运算,这使得很难应用对数方法。一个常用的解决办法是使用辅助规范系数,它的值随着时间 t 增加,因此乘以了规范系数的概率总是保持在计算机的浮点数的界内。在每次迭代结束时,如果参数被重估,那么这些规范系数将会被取消。详细的讨论以及其他实现的具体问题可以参见 (Levinson et al. 1983), (Rabiner and Juang 1993:365 ~ 368) (Cutting et al. 1991) 以及 (Dermatas and

Kokkinakis 1995)。尽管有时候也需要进行加法运算,但是主要的替换方式仍然是对数方法,而比较有效的做法是在每次进行加法的时候计算合适的规范因子:

```

funct log_add =
  if ( $y - x > \log big$ )
    then  $y$ 
  elseif ( $x - y > \log big$ )
    then  $x$ 
  else  $\min(x, y) + \log(\exp(x - \min(x, y)) + \exp(y - \min(x, y)))$ 
fi.

```

(9.21)

其中 big 是一个足够大的常数,比如 10^{30} 。对于一个类似这样进行大量数值计算的算法,我们也需要留心舍入错误,这些问题超出了本章所要讨论的内容。

9.4.2 变形

HMM 存在很多变形形式,类似于有限状态自动机。一个 HMM 允许弧转移发生而不发射符号,这称为 ϵ 或空转移(Bahl et al.1983)。另一个常用的变形是使得输出分布仅仅依赖于一个单一状态,而不是当扫过一条弧时被弧连接的两个状态,这样就有效地描述了软饮料机的情况。在这个模型下,我们可以把输出看成是状态选择函数,而不是经过的弧的函数。输出是状态选择函数的模型,实际上在自然语言处理中应用得很广泛,因为它很自然地对应了一个词性标注模型,这个问题在第10章中也会看到。实际上,人们也许会认为我们过分强调了这一章中的弧发射模型。但是,我们选择弧发射模型是因为用它来模拟状态发射模型很容易,然而用状态发射模型来模拟弧发射模型非常困难。如上所述,并不需要考虑从状态得到输出的简单模型,而是仍然把输出看成是由弧得到的,但是输出分布恰巧对所有从特定节点出发(如果愿意,也可以以特定节点结束)的弧都是一样的。

这里提出了一个一般化策略。一个 HMM 模型问题是需要估计定义模型的大量参数的问题,如果没有足够的数据,要做出精确的估计是不可能的。处理这种情况的一个直接策略是引入一个假设,即在特定的弧或者特定状态上的概率分布彼此相同。这就涉及到参数尝试的问题,由此我们可以得到限制性状态和限制性弧。另一个减少模型参数数目的方法是确定某些东西是不可能的(例如,它们具有零概率),因此可以把零化结构引入模型。把一些东西设成不可能即可把很多结构加入到模型中,因而可以极大地提高参数重估算法的性能,但这仅在某些情况下是适用的。

9.4.3 多输入观测序列

我们已经详细讨论了如何处理单输入序列的算法。但是怎样在多输入序列上进行训练呢?对于我们假设的 HMM 模型,每个状态都与其他状态相连(具有非零转移概率,有时称为遍历模型),有一个简单的解决办法。我们只需要把所有的观测序列连接在一起,把它们当成一个长输入进行训练。惟一的不足之处是,我们没有足够的数据来有效地重估初始概率。但是,人们也经常使用非全连接的 HMM 模型。例如,人们有时使用包含一个有序状态集的前馈模型(feed forward model),并且每次只能前进到相同或者更高数字的状态。如果 HMM 不是全连接的(包含零化结构),或者我们希望能够重估初始概率,就需要扩展重估公式来处理输入序

列。假设输入是互相独立的,那么这样就是很直接的。这里并不列出这个公式,但在 11.3.4 节中会列出类似的公式。

9.4.4 参数值的初始化

重估过程只能保证我们找到一个局部极值。如果我们想要找到全局极值,则要尽量使得 HMM 在全局极值附近的参数空间开始。我们可以通过粗略估计参数的最佳值来实现这一点,而不是随机地设定其值。在实践中,好的初始估计对于输出参数 $B = \{b_{jk}\}$ 是非常重要的,随机估计 A 和 π 的初始参数通常就够了。

9.5 深入阅读

Viterbi 算法最开始出现在(Viterbi 1967)中。隐马尔可夫模型的数学思想是由 Baum 以及他的同事在 20 世纪 60 年代末 70 年代初提出来的(Baum et al. 1970),而且被来自防御分析学院的 Jack Ferguson 应用在了语音识别上。在 20 世纪 70 年代被 CMU 的 Baker(Baker 1975)以及 IBM 的 Jelinek 和他的同事(Jelinek et al. 1975; Jelinek 1976)应用在了语音处理上,之后被 IBM 应用在了其他语言模型上,例如词性标注模型。

关于 HMM 模型的参考文献(在语音识别领域)有很多,包括(Levinson et al. 1983; Knill and Young 1997; Jelinek 1997)。最著名的是(Rabiner 1989; Rabiner and Juang 1993),其中考虑了连续 HMM(输出是实数)以及我们这里考虑的离散 HMM,包含语音识别中 HMM 应用的信息,同时也可以用作 HMM 开发与应用的相当全面的参考。我们对 HMM 的描述很接近 Paul(1990)。

在这一章中,我们提出了一个固定的 HMM 结构,而且也只涉及了在这个结构下学习最优的参数。但是,面对一个新问题时我们应该选取什么样大小和形式的 HMM 呢?有时问题的性质决定了结构,比如我们将在下一章中讨论的使用 HMM 进行标注的应用。如果不是这种情况,就需要在某种原则下学习一个合适的 HMM 模型,这个原则是能够充分描述数据的最简洁的 HMM 模型,参见(Stolcke and Omohundro 1993)。

在生物信息学方面,HMM 被广泛用来分析基因序列。参见实例(Baldi and Brunak 1998; Durbin et al. 1998)。作为语言学家,我们会发现很难处理一些建立在四个符号上的字母表的问题,但是生物信息学领域非常适合应用一些隐马尔可夫模型的新技术。

9.6 习题

- 9.1 [★]为表 4.2 中的一种电话号码类型构造一个类似于图 9.1 的马尔可夫模型。
- 9.2 [★]如果机器总是从冰茶状态开始,那么看到的输出序列是{col, lem}的概率是多少?
- 9.3 [★]证明式(9.9)的计算方法需要做 $(2T+1) \cdot N^{T+1}$ 次乘法。
- 9.4 [★]如果我们继续式(9.20)描述的训练,即在原 HMM 和原训练序列上继续运行 Baum-Welch 算法,每个参数会达到什么样的极限值呢?为什么?
Baum-Welch 算法在这里表现得如此奇怪的原因很明显:训练序列太短以至于不能准确

地表达软饮料机的行为。

- 9.5 [★]我们注意到在 π 中是0的参数依然是0。这是偶然的吗？如果我们再进行一次 Baum-Welch 重估迭代,那么 B 中为0的参数的值会发生什么样的变化？对于0参数的 Baum-Welch 重估,我们可以得出什么样的一般化结论？

第 10 章 词 性 标 注

自然语言处理研究的最终目标是分析和理解语言。在前面的章节中已经看到,我们离实现这个目标还很遥远。由于这个原因,很多 NLP 的研究都集中在一些所谓的中间任务上,即在不需完全理解语言的情况下如何了解语言的内在结构。词性标注,或简称为标注,便是其中之一。标注的任务是为句子中的每个词都标上一个合适的词性,也就是我们要确定每个词是名词、动词、形容词或其他词性。为了给读者一个直观的认识,下面是一个词性标注的例子:

The-AT representative-NN put-VBD chairs-NNS on-IN the-AT table-NN. (10.1)

表 10.1 中列出了本章所用到的词性标记,它们一般遵循 Brown/Penn 标注集的标准(参见 4.3.2 节)。另外需要注意的是,对于同样的句子还可能有其他标注(例如一种很不常见的用法,put 用做 *an option to sell* 的意思):

The-AT representative-JJ put-NN chairs-VBZ on-IN the-AT table-NN. (10.2)

表 10.1 英语标注中常用的一些词性

标记	词性
AT	冠词
BEZ	词 <i>is</i>
IN	介词
JJ	形容词
JJR	形容词比较级
MD	情态动词
NN	单数或者物质名词
NNP	单数或者专有名词
NNS	复数名词
PERIOD	. : ? !
PN	人称代词
RB	副词
RBR	副词比较级
TO	词 <i>to</i>
VB	动词原形
VBD	动词过去式
VBG	动词现在分词,动名词
VTN	动词过去分词
VBP	动词非第三人称单数现在时
VBZ	动词第三人称单数现在时
WDT	<i>wh</i> -限定词(<i>what, which</i>)

但是,这种标注方法会造成阅读上的语义不连贯。而且,它在语法上也是不太可能的,因为 *put* 用做名词并且 *chairs* 用做不及物动词的情况比较少见。

这个例子表明标注是一种有限语法消歧问题。许多词都不只有一种语法类别,即存在兼类的情况。在进行标注时,要努力确定对于句子中的一个词的特定用法,最可能是哪种语法类别。

标注是一个有限范围的问题。与构造一个完整的句法分析不同,我们只是为句子中的词选定一个语法类别。例如,我们并不考虑为介词短语找到正确的附着。一般来说,标注要比句法分析容易得多,而且准确率也很高。在最成功的方法中,96%~97%的词语歧义能够被正确地消歧。但是,必须认识到这么高的精确率数字并没有看上去那么好,因为它是以每个词为基础进行评估的。例如,在很多文本(诸如报纸)中,每个句子都有超过20个词,在这样的句子中,即使达到了96%的标注正确率,也意味着平均每个句子都会有一个词被标注错误。

即使很有限,我们从标注中得到的信息还是很有用的。标注用途很广,例如可以被应用于信息抽取、问题回答以及浅层句法分析等方面。按照 Francis 和 Kučera 的观点(他们曾经于20世纪60年代至20世纪70年代在 Brown 大学进行语料库语言学工作),标注是一个比完整句法分析更有用、更容易控制的中间层表达(Francis and Kučera 1982)。

下面我们将要介绍马尔可夫模型标注器、隐马尔可夫模型标注器以及基于转换的标注。在本节的最后,我们讨论不同标注方法的准确率水平,这里首先应该对标注所需的信息类别做一些一般性的注解。

10.1 标注中的信息源

怎样才能确定文本中一个词的词性是否正确呢?有两个基本的信息源。一个信息源是观察我们感兴趣的词邻近上下文中的其他词的标注。这些词的词性也可能是有歧义的,但是这里有一些基本的观测准则:很多词性序列都是很常见的,例如 AT JJ NN,然而有些词性序列则基本上不可能,例如 AT JJ VBP。因此,当确定短语 *a new play* 中的词 *play* 的词性是 NN 还是 VBP 时,很明显我们会选择前者。这种类型的结构语段信息是标注中最明显的信息源,但是,就其本身而言却不是很成功。例如,在 Greene and Rubin(1971)中,早期的使用结构语段信息的基于确定规则的标注器,其正确率只有77%。似乎这使得标注问题看上去显得很困难,其中的一个原因在于英语中的实词有很多词性。例如,在英语中任意名词可以转化为动词,比如, *Next, you **flour** the pan* 或 *I want you to **web** our annual report*。这就意味着几乎所有的名词也需要作为动词列在词典中,与此同时我们也丢弃了很多标注需要的约束信息。

第二个信息源是利用词本身提供的信息。例如,虽然 *flour* 可以被用做动词,但是它更多是作为名词出现的。Charniak et al. (1993)对如何利用这些信息做了总结性的阐述,设计了一个 dumb 标注器,简单地把最常用的标注分配给每个词,结果达到了令人惊奇的水平:90%^①。这又使得标注看上去十分容易,即至少给出了一种比较好的情况,我们将在后面讨论这个问题。这个 dumb 标注器的性能被用做后续研究的基准性能(baseline)。现代的标注器都在某些方面结合使用了结构语段信息(查看标注序列的信息)和词汇信息(根据考虑的词来预测标注)。

① 这种方法的有效性早先由 Atwell(1987)提及。

词汇信息是很有用的,因为词语的不同词性的使用分布极不均匀。即使是对于有兼类情况的词语,大多数情况下它们只作为一种特殊词性使用。实际上,通常某种词性会被作为基本词性,其他的词性由它派生出来。在传统语法中,我们经常看到文本中的一个词被归类为用做形容词的名词,这就混淆了词的基本词性与其用在当前上下文中的词性。在这一章中,正如现代语言学的一般情况,我们将考虑确定后一个概念,即当前上下文中的词性,与此同时词性间词语的分布也提供了大量的附加信息。实际上,不均匀分布也是导致我们预期用统计方法进行标注比用确定方法好的一个原因。在确定方法中我们只能说一个词或者是或者不是动词,如果很不常见,便会趋向于丢弃动词的可能性(因为这么做可以提高整体性能水平),然而在统计方法中,一个词既可以是名词,也可以是动词或者其他词性的词,只不过是名词的可能性要远远大于其他词性。因此,在语法结构消歧处理中,允许定量信息的框架方法比纯符号方法能更充分地代表语言学知识。

10.2 马尔可夫模型标注器

10.2.1 概率模型

在马尔可夫模型标注中,我们把文本中的标记序列看成是一条马尔可夫链。如第9章中所讨论的,马尔可夫链有下面两个属性:

- 有限视野 (limited horizon) $P(X_{i+1} = t^j | X_1, \dots, X_i) = P(X_{i+1} = t^j | X_i)$
- 时间不变性 (stationary) $P(X_{i+1} = t^j | X_i) = P(X_2 = t^j | X_1)$

我们假设一个词语的标记只依赖于前面的标记(有限视野),而且这种依赖性并不随时间而改变(时间不变性)。例如,如果在句子的开始一个限定动词出现在一个代词之后的概率为0.2,那么这个概率在我们标注句子的其余部分(或者是新句子)时也不会改变。对于大多数概率模型,这两个属性只是对现实的一种近似。例如,有限视野属性并不能把类似 *wh-* 抽取中的远距离关系模型化,这实际上也是 Chomsky 反对马尔可夫模型用在自然语言处理中的最主要的依据。

根据(Charniak et al. 1993),我们使用表10.2中的符号。我们用下标指示标注语料库和句子中特殊位置的词和词性,用上标表示词典中的词和标注集中的词性类别。在这个简洁的符号表示中,我们可以把上述有限视野属性表示成下面的形式:

$$P(t_{i+1} | t_{1,i}) = P(t_{i+1} | t_i)$$

我们使用一个手工标注的文本组成的训练集来学习词性标记序列规则。标记 t^k 跟随 t^j 的最大似然估计来自于不同的标记跟随某个特定标记的相对频率估计,如下所示:

$$P(t^k | t^j) = \frac{C(t^j, t^k)}{C(t^j)}$$

例如,查看例子中是怎样标注 *a new play* 的,我们预期会得到 $P(\text{NN} | \text{JJ}) \gg P(\text{VBP} | \text{JJ})$ 。实际上,在 Brown 语料库中, $P(\text{NN} | \text{JJ}) \approx 0.45$, 而 $P(\text{VBP} | \text{JJ}) \approx 0.0005$ 。

表 10.2 进行标注时的惯用符号

w_i	语料库位置 i 处的词语
t_i	w_i 的标记
$w_{i:i+m}$	位置 i 与 $i+m$ 之间出现的词语(或符号: $w_i \cdots w_{i+m}, w_i, \cdots, w_{i+m}, w_{i(i+m)}$)
$t_{i:i+m}$	$w_i \cdots w_{i+m}$ 的标记 $t_i \cdots t_{i+m}$
w^l	词典中的第 l 个词
t^j	标注集中的第 j 个标记
$C(w^l)$	w^l 在训练集中出现的次数
$C(t^j)$	t^j 在训练集中出现的次数
$C(t^j, t^k)$	t^k 跟随 t^j 之后在训练集中出现的次数
$C(w^l; t^j)$	w^l 被标注成 t^j 在训练集中出现的次数
T	标注集中标记的个数
W	词典中词的个数
n	句子长度

通过估计概率 $P(t_{i+1} | t_i)$, 我们能够计算一个特定标记序列的概率。在实践中, 其任务是找到一个词序列最可能的标记序列(因为这里的马尔可夫模型的状态是标记)。我们把马尔可夫模型每次离开一个状态时发射的词连接起来。这与第9章中 HMM 的符号发射概率 b_{ijk} 是类似的:

$$P(O_n = k | X_n = s_i, X_{n+1} = s_j) = b_{ijk}$$

其区别是, 如果有一个标注过的语料库, 我们可以直接查看状态(或标记)。每个标记对应一个不同的状态。我们也可以通过最大似然估计来直接估计一个词被一个特定的状态(标记)发射出来的概率:

$$P(w^l | t^j) = \frac{C(w^l, t^j)}{C(t^j)}$$

现在我们已经为寻找一个句子 $w_{1,n}$ 的最佳标记序列 $t_{1,n}$ 做好了准备。应用贝叶斯法则, 我们可以写成:

$$\begin{aligned} \arg \max_{t_{1,n}} P(t_{1,n} | w_{1,n}) &= \arg \max_{t_{1,n}} \frac{P(w_{1,n} | t_{1,n}) P(t_{1,n})}{P(w_{1,n})} \\ &= \arg \max_{t_{1,n}} P(w_{1,n} | t_{1,n}) P(t_{1,n}) \end{aligned} \quad (10.3)$$

现在可以把这个表达式简化成从语料库中通过估计能直接得到的参数。除了有限视野假设式(10.5), 我们还对词语做了两个假设:

- 词语之间是独立的, 见式(10.4)
- 词语的出现只依赖于它本身的标注, 见式(10.5)

$$P(w_{1,n}|t_{1,n})P(t_{1,n}) = \prod_{i=1}^n P(w_i|t_{1,n}) \quad (10.4)$$

$$\times P(t_n|t_{1,n-1}) \times P(t_{n-1}|t_{1,n-2}) \times \cdots \times P(t_2|t_1)$$

$$= \prod_{i=1}^n P(w_i|t_i) \quad (10.5)$$

$$\times P(t_n|t_{n-1}) \times P(t_{n-1}|t_{n-2}) \times \cdots \times P(t_2|t_1)$$

$$= \prod_{i=1}^n [P(w_i|t_i) \times P(t_i|t_{i-1})] \quad (10.6)$$

我们定义 $P(t_1|t_0) = 1.0$ 来简化我们的表达式。

因此,确定一个句子的最优标注的等式是:

$$\hat{t}_{1,n} = \arg \max_{t_{1,n}} P(t_{1,n}|w_{1,n}) = \arg \max_{t_{1,n}} \prod_{i=1}^n P(w_i|t_i)P(t_i|t_{i-1}) \quad (10.7)$$

这个训练马尔可夫模型标注器的算法总结在图 10.1 中。根据表 10.3 和表 10.4,读者可做习题 10.4 和习题 10.5。下一节讲述怎样用训练好的马尔可夫模型标注器来进行标注。

```

1 for all tags  $t^j$  do
2   for all tags  $t^k$  do
3      $P(t^k|t^j) := \frac{C(t^j, t^k)}{C(t^j)}$ 
4   end
5 end
6 for all tags  $t^j$  do
7   for all words  $w^l$  do
8      $P(w^l|t^j) := \frac{C(w^l, t^j)}{C(t^j)}$ 
9   end
10 end
```

图 10.1 训练显马尔可夫模型标注器的算法。在大多数的实现中,一些平滑思想被用来估计 $P(t^k|t^j)$ 和 $P(w^l|t^j)$

表 10.3 Brown 语料库中一些标记转移的理想计数。例如,NN在AT后面出现了48636次

第一个标记	第二个标记					
	AT	BEZ	IN	NN	VB	PERIOD
AT	0	0	0	48 636	0	19
BEZ	1973	0	426	187	0	38
IN	43 322	0	1325	17 314	0	185
NN	1067	3720	42 470	11 773	614	21 392
VB	6072	42	4758	1476	129	1522
PERIOD	8016	75	4656	1329	954	0

表 10.4 Brown 语料库中一些词和标记同时出现的理想计数。例如, move 被标注成 NN 一共出现了 36 次

	AT	BEZ	IN	NN	VB	PERIOD
<i>bear</i>	0	0	0	10	43	0
<i>is</i>	0	10 065	0	0	0	0
<i>move</i>	0	0	0	36	133	0
<i>on</i>	0	0	5484	0	0	0
<i>president</i>	0	0	0	382	0	0
<i>progress</i>	0	0	0	108	4	0
<i>the</i>	69 016	0	0	0	0	0
	0	0	0	0	0	48 809

10.2.2 Viterbi 算法

我们可以对一个长度为 n 的句子的所有可能标记 $t_{1,n}$ 用式(10.7)进行评估,但是这会使得标注算法的复杂度是句子长度的指数倍。一个有效的标注算法是第 9 章中提出的 Viterbi 算法。回顾一下, Viterbi 算法有三步: (i) 初始化; (ii) 推导; (iii) 终止和路径读出。我们计算了两个函数: $\delta_i(j)$ 给出了词 i 处于状态 j (标注为 j) 时的概率; $\psi_{i+1}(j)$ 给出了词 $i+1$ 处于状态 j 时词 i 所处的最可能状态(或标记)。读者在继续阅读之前可以回顾一下 9.3.2 节中关于 Viterbi 算法的讨论。本章中我们会用状态表示词性标记, 因为模型中的状态对应词性标记。但是要注意, 这仅适用于二元标注器。

算法的初始化步骤把概率 1.0 分配给标记 PERIOD^①:

$$\delta_1(\text{PERIOD}) = 1.0$$

$$\delta_1(t) = 0.0 \quad \text{当 } t \neq \text{PERIOD}$$

即, 句子由句点分开, 为了方便起见我们假设在文本中的第一个句子之前有一个句点。

推导步骤基于式(10.7), 其中 $a_{jk} = P(t^k | t^j)$, $b_{jw^i} = P(w^i | t^j)$

$$\delta_{i+1}(t^j) = \max_{1 \leq k \leq T} [\delta_i(t^k) \times P(w_{i+1} | t^j) \times P(t^j | t^k)], \quad 1 \leq j \leq T$$

$$\psi_{i+1}(t^j) = \arg \max_{1 \leq k \leq T} [\delta_i(t^k) \times P(w_{i+1} | t^j) \times P(t^j | t^k)], \quad 1 \leq j \leq T$$

最后, 终止和路径读出步骤如下, 其中 $X_1, \dots, X_n$ 是我们为词语序列 $w_1, \dots, w_n$ 选择的标记:

$$X_n = \arg \max_{1 \leq j \leq T} \delta_n(t^j)$$

$$X_i = \psi_{i+1}(X_{i+1}), \quad 1 \leq i \leq n-1$$

$$P(X_1, \dots, X_n) = \max_{1 \leq j \leq T} \delta_{n+1}(t^j)$$

如何用显马尔可夫模型标注器进行标注在图 10.2 中做了描述。

① 根据本书配套网站上提供的勘误表, 已有人指出, 下式以及图 10.2 的计算中都将句点作为句子的开始, 但文中其他地方并不是这样叙述的。——译者注

```

1 comment: Given: a sentence of length  $n$ 
2 comment: Initialization
3  $\delta_1(\text{PERIOD}) = 1.0$ 
4  $\delta_1(t) = 0.0$  for  $t \neq \text{PERIOD}$ 
5 comment: Induction
6 for  $i := 1$  to  $n$  step 1 do
7   for all tags  $t^j$  do
8      $\delta_{i+1}(t^j) := \max_{1 \leq k \leq T} [\delta_i(t^k) \times P(w_{i+1}|t^j) \times P(t^j|t^k)]$ 
9      $\psi_{i+1}(t^j) := \arg \max_{1 \leq k \leq T} [\delta_i(t^k) \times P(w_{i+1}|t^j) \times P(t^j|t^k)]$ 
10  end
11 end
12 comment: Termination and path-readout
13  $X_{n+1} = \arg \max_{1 \leq j \leq T} \delta_{n+1}(j)$ 
14 for  $j := n$  to 1 step -1 do
15    $X_j = \psi_{j+1}(X_{j+1})$ 
16 end
17  $P(X_1, \dots, X_n) = \max_{1 \leq j \leq T} \delta_{n+1}(t^j)$ 

```

图 10.2 用显马尔可夫模型标注器进行标注的算法

术语注解:马尔可夫模型和隐马尔可夫模型。读者会注意到,为了实现标注目的,本章中马尔可夫模型被当成隐马尔可夫模型对待。这是因为在训练中我们可以观测马尔可夫模型的状态(标注语料库的标记),但是在应用马尔可夫模型进行标注时只观测词语。我们可以说使用马尔可夫模型标注的形式实际上是一种混合形式。我们在训练中构造了“显”马尔可夫模型,在使用它们标注新语料库的时候又把它们当成了隐马尔可夫模型。

10.2.3 算法的变形

未登录词

我们已经说明了怎样估计语料库中出现过的词语的生成概率,但有时候需要标注的句子中的一些词并没有出现在训练语料库中,有些词甚至不在词典里。前面讨论过,一个词标注的先验分布(或者至少是一个词最常用的标注)在解决标注问题时起了很大的作用。这就意味着未登录词是标注器的一个主要问题。在实践中,不同标注器在不同语料库上的不同准确率主要是由未登录词的比例决定的,智能被用到标注器中,从而允许标注器尝试为未登录词猜测一个词性标记。

未登录词的最简单的模型是假设它们可以有任意词性(也许只是一些开放的词性类别,即名词或动词等,但不是介词或冠词)。未登录词对应整个词典被给予了一个词性分布。尽管这种方法在某些情况下是可行的,但是因为丢失了这些词语的词汇信息,所以在很大程度上降低了标注器的准确率,因此人们尝试利用词语的其他特征以及上下文来提高对未登录词的词汇概率估计。一般我们可以使用变形或者其他线索来推导词语的可能词性。例如,以-ed结尾的词可能是过去式或过去分词。Weischedel et al. (1993)基于三种信息估计了词语生成概率:一个标注可以生成一个未登录词的可能性有多大(这个概率对于某些标注是0,例如PN,人称代词);生成大写词或者小写词的可能性;生成连字符或者特殊后缀的可能性:

$$P(w^l|t^j) = \frac{1}{Z} P(\text{unknown word}|t^j)P(\text{capitalized}|t^j)P(\text{endings/hyph}|t^j)$$

其中, Z 是一个归一化常数。这个模型把未登录词的错误率从 40% 以上降低到了 20% 以下。

Chamiak et al. (1993) 提出了另外一种模型, 它同时依赖词根和后缀, 并且能够从多种变形分析中进行选择, 例如 *do-es* (动词形式) 和 *doe-s* (名词的复数形式)。

很多关于未登录词方面的工作都假设了特征之间的独立性, 但是独立性并不是一个好的假设。例如, 大写词很有可能是未登录词, 于是 Weischedel et al. (1993) 的模型中特征“未登录词”和“大写”实际上并不互相独立。Franz (1996; 1997) 利用依存关系构建了一个未登录词模型, 提出了一个描述主要影响(特征本身的影响)和相互影响(比如“未登录词”和“大写”之间的依存关系)的对数线性模型。基于贝叶斯法则的一个方法参见 Samuelsson (1993)。

表 10.5 标注中处理未登录词的概率表。例如, $P(\text{unknown word} = \text{yes} | \text{NNP}) = 0.05$, $P(\text{ending} = -ing | \text{VBG}) = 1.0$

特征	值	NNP	NN	NNS	VBG	VBZ
未知词	yes	0.05	0.02	0.02	0.005	0.005
	no	0.95	0.98	0.98	0.995	0.995
大写词	yes	0.95	0.10	0.10	0.005	0.005
	no	0.05	0.90	0.90	0.995	0.995
结尾	-s	0.05	0.01	0.98	0.00	0.99
	-ing	0.01	0.01	0.00	1.00	0.00
	-tion	0.05	0.10	0.00	0.00	0.00
	other	0.89	0.88	0.02	0.00	0.01

三元语法标注器

基本的马尔可夫模型标注器可以通过好几种方式扩展。在构建模型方面, 到目前为止我们一般采用前面的标记进行预测, 称为二元语法标注器, 因为我们考虑的基本单元是前面的标记和当前标记。我们可以把标注当成是选取最可能的二元语法问题。

如果考虑了更多的上下文, 预期会有更精确的预测。例如, 标注 RB (副词) 可以在一个动词的过去式 (VBD) 或过去分词 (VBN) 前面。因此, 类似 *clearly marked* 的词语序列在只记住前面一个标记的马尔可夫模型中是先天性歧义的。三元语法标注器记忆了两个标记, 因此能够让我们区分更多的情况。例如, *is clearly marked* 和 *he clearly marked* 分别表示了 VBN 和 VBD, 因为三元组 BEZ RB VBN 比 PN RB VBN 更常见。Church (1988) 中描述了一个三元语法标注器, 它也许是在标注领域中被引用最多的资料, 而且引发了很多 NLP 研究者对于词性标注的兴趣。

插值和可变记忆

基于更长历史的预测并不总是一个好办法。例如, 以逗号分隔的词语之间通常没有近距离的语法结构依存关系。因此, 我们知道逗号之前的词性对于确定逗号之后的正确标注并没有多少帮助。实际上, 三元语法标注器因为数据稀疏问题在某些情况下会比二元语法标注器做出更糟糕的预测。三元转移概率是基于不常见事件的估计, 于是得到一个糟糕估计的可能性就会大大增加。

解决这个问题的一个办法是采用一元、二元和三元概率的线性插值:

$$P(t_i|t_{1,i-1}) = \lambda_1 P_1(t_i) + \lambda_2 P_2(t_i|t_{i-1}) + \lambda_3 P_3(t_i|t_{i-1}, t_{i-2})$$

这种线性插值方法已在第 6 章中讲过,怎样使用 HMM 来估计参数 λ_i 已在第 9 章中讲过。

一些研究人员基于错误率分析和先验语言学知识有选择地扩充了低阶马尔可夫模型。例如, Kupiec(1992b)观测到一个一阶 HMM 系统把序列 *the bottom of* 错误标注成了 AT JJ IN, 然后用一个特殊的网络为这种结构构造扩展了一阶 HMM, 因此可以学习到, 介词不可能出现在 AT JJ 中的序列后面。这种方法等同于手工地为—阶记忆不充足的情况选择了一个高阶状态。

一个相关的方法是可变记忆马尔可夫模型(Variable Memory Markov Model, VMMM)(Schvitz and Singer 1994)。VMMM 用混合长度的状态代替了二元或者三元语法标注器中的固定长度状态。一个 VMMM 标注器可以从一个记忆了前两个标记(对应三元语法模型)的状态转移到一个记忆了前三个标记(对应四元语法模型)的状态, 再转移到一个没有记忆(对应一元语法模型)的状态。一个特定序列记忆的符号数目是在训练中基于信息论的准则决定的。对比线性插值, VMMM 有条件地使用记忆长度进行预测, 取代了对于所有的序列进行固定加权和方法。VMMM 是通过自顶向下划分状态来构造的。另一种方法是通过模型合并自底向上地构造这类模型(Stolcke and Omohundro 1994a; Brants 1998)。

层次化非发射马尔可夫模型(Hierarchical non-emitting Markov model)是由 Ristad and Thomas (1997)提出的一个更为强大的模型。通过引入非发射转移(不发射词语或发射空词语 ϵ 的状态之间的转移), 这个模型可以存储任意长度距离的状态之间的依存性。

平滑

线性插值是平滑估计的一种方法。我们可以使用第 6 章中讨论的任何其他估计方法来进行平滑。例如, Charniak et al.(1993)使用了类似于加 1 法的一种思想(但是要注意, 通常来说, 它并不给出一个严格意义上的概率分布...):

$$P(t^j|t^{j-1}) = (1 - \epsilon) \frac{C(t^{j-1}, t^j)}{C(t^{j-1})} + \epsilon$$

平滑词语的生成概率比平滑转移概率更重要, 因为有很多在训练语料库中没有出现的稀少词。这里加 1 法也得到了应用(Church 1988)。Church 对每个特定词在词典中列出的所有词性的数目都加 1, 这样就保证了对 w^l 而言所有列出的可能词性 t^j 都有一个非零概率:

$$P(t^j|w^l) = \frac{C(t^j, w^l) + 1}{C(w^l) + K_l}$$

其中 K_l 是 w^l 可能的词性的数目。

可逆性

我们已经描述了一个从左到右译码(标注)的马尔可夫模型。实际上, 从右到左译码是等同的。下面的推导说明了其中的原因:

$$\begin{aligned} P(t_{1,n}) &= P(t_1)P(t_{1,2}|t_1)P(t_{2,3}|t_2) \dots P(t_{n-1,n}|t_{n-1}) \\ &= \frac{P(t_1)P(t_{1,2})P(t_{2,3}) \dots P(t_{n-1,n})}{P(t_1)P(t_2) \dots P(t_{n-1})} \\ &= P(t_n)P(t_{1,2}|t_2)P(t_{2,3}|t_3) \dots P(t_{n-1,n}|t_n) \end{aligned} \quad (10.9)$$

如果初始状态和结束状态的概率是一样的(标注便属于这种情况,因为都对应标注 PERIOD),那么前向概率和后向概率都是一样的,所以我们选择哪个方向无关紧要。这里描述的标注器从左到右移动。Church 的标注器采用了相反的方向。

最大似然性:状态(标记)序列

正如我们在第9章中指出的,Viterbi 算法可以找到最可能的状态(或标记)序列,即最大化 $P(t_{1,n} | w_{1,n})$ 。我们也可以对所有的 i 最大化 $P(t_i | w_{1,n})$,等同于对不同的标注序列求和。

作为一个例子,我们考虑例句(10.10):

Time flies like an arrow. (10.10)

根据从训练语料库中获得的转移概率,我们假设序列(10.11a)和序列(10.11b)是可能相对较大的标记序列(假设概率为 0.01),序列(10.11c)是可能较小的标记序列(假设概率为 0.001),序列(10.11d)是绝对不可能的,因为转移概率 $P(VB|VBZ)$ 是 0.0。

- | | |
|---------------------|--------------------|
| a. NN VBZ RB AT NN. | $P(\cdot) = 0.01$ |
| b. NN NNS VB AT NN. | $P(\cdot) = 0.01$ |
| c. NN NNS RB AT NN. | $P(\cdot) = 0.001$ |
| d. NN VBZ VB AT NN. | $P(\cdot) = 0$ |
- (10.11)

对于这个例子,我们会发现标记序列(10.11a)和标记序列(10.11b)都是最大可能的序列 $P(t_{1,n} | w_{1,n})$ 。但是,如果我们对于所有 i 最大化 $P(t_i | w_{1,n})$,会获得标记序列(10.11c)。这是因为 $P(X_2 = NNS | \text{Time flies like an arrow}) = 0.011 = P(b) + P(c) > 0.01 = P(a) = P(X_2 = VBZ | \text{Time flies like an arrow})$,以及 $P(X_3 = RB | \text{Time flies like an arrow}) = 0.011 = P(a) + P(c) > 0.01 = P(b) = P(X_3 = VB | \text{Time flies like an arrow})$ 。

Merialdo(1994:164)设计的实验表明,最大化单个标记和标记序列的可能性之间在准确率上并没有很大差别。从直观上讲这是很好理解的。使用 Viterbi,标记转移可能是最敏感的,但是如果有什么地方出错了,我们有时候会得到一个几处标记错误的序列;虽然一个标记跟着一个标记,但是一个错误不会影响其他词的标记,因此更容易得到偶然的分散错误。在实际应用中,不连贯起来求概率最大的序列(类似上面的 NN NNS RB AT NN)是没什么用的,因此 Viterbi 算法是马尔可夫模型比较偏好的标注方法。

10.3 隐马尔可夫标注器

在我们有很大的标注语料库的时候,马尔可夫模型标注器可以工作得很好,但通常不是这种情况。我们也许会希望标注一个特定领域内的文本,这个领域内的词语生成概率与可获得的训练文本是不一样的。或者,我们需要标注一个外国语言的文本,但是根本就没有训练语料库存在。

10.3.1 隐马尔可夫模型在词性标注中的应用

如果我们没有训练数据,也可以使用 HMM 来学习标记序列的规则。我们知道第9章中介绍的 HMM 包含如下元素:

- 一个状态集
- 一个输出字母表
- 初始状态概率
- 状态转移概率
- 符号发射概率

和显马尔可夫模型的例子相同,状态对应标记。输出字母表包含词典中的词语或词语类别。

我们随机地初始化 HMM 的所有参数,但这会使得标注问题变得太不受限制。通常情况下词典信息被用来限制模型参数。如果输出字母表包含词语,如果对应的词语-标记对在词典中列出(例如, JJ 并没有被列为 *book* 的一个可能的词性),我们把词语生成(=符号发射)概率设成 0。另一种方法是,我们可以把词语聚集到词语等价类中,使得所有同一类的词语允许同样的标注。例如,我们可以把 *bottom* 和 *top* 聚集到类 JJ-NN 中(如果它们都只列出两个词性, JJ 和 NN)。第一种方法是由 Jelinek(1985)提出的,第二种方法是由 Kupiec(1992b)提出的。我们用 $b_{j,l}$ 记录词语(或词类) l 由标记 j 发射的概率。这就意味着与显马尔可夫模型类似,一个标记的输出并不依赖于下一个标记(状态)。

● Jelinek 的方法

$$b_{j,l} = \frac{b_{j,l}^* C(w^l)}{\sum_{w^m} b_{j,m}^* C(w^m)}$$

其中,求和是针对词典中的所有词 w^m 的,而且

$$b_{j,l}^* = \begin{cases} 0 & \text{如果 } t^j \text{ 不是 } w^l \text{ 所允许的词性} \\ \frac{1}{T(w^l)} & \text{其他} \end{cases}$$

其中 $T(w^l)$ 是 w^l 可能的标记数目。

Jelinek 的方法等同于用 $P(w^l | t^j)$ 的最大似然估计初始化 HMM,假设词与它的每个可能的标记同时出现的概率是相等的。

● Kupiec 的方法

首先,把所有具有相同可能词性集合的词语聚集到‘metawords’ u_L 中。这里, L 是整数 1 到 T 的子集,其中 T 是标记集中不同标记的数目:

$$u_L = \{w^l | j \in L \leftrightarrow t^j \text{ 是 } w^l \text{ 所允许的} \} \quad \forall L \subseteq \{1, \dots, T\}$$

例如,如果 $NN = t^5$ 而且 $JJ = t^8$,那么 $u_{\{5,8\}}$ 就包含了词典中所有只允许标注 NN 和 JJ 的词语。

接下来我们采用 Jelinek 方法中处理词语的方式来处理这些‘metawords’ u_L ^①:

① Kupiec 实际使用的初始化是我们这里展示的方法的一种变形。我们尽量使得 Jelinek 和 Kupiec 的方法之间的相似性更加明晰。

$$b_{j,L} = \frac{b_{j,L}^* C(u_L)}{\sum_{u_L'} b_{j,L'}^* C(u_L')}$$

其中, $C(u_L')$ 是 u_L' 中的词出现的次数, 分母中的和是基于所有 'metawords' u_L 的, 而且

$$b_{j,L}^* = \begin{cases} 0 & \text{如果 } j \notin L \\ \frac{1}{|L|} & \text{其他} \end{cases}$$

其中, $|L|$ 是 L 中索引的数目。

Kupiec 方法的优点在于我们并不需要为每个词单独地调整参数集合。通过引入等价类, 总的参数数目大大减少了, 因此这个小的集合更容易被可靠地估计。但是, 如果有足够的训练语料像 Jelinek 方法所做的那样来精确地一个词一个词地估计参数, 等价类的优势也将不复存在。Merialdo(1994) 的实验表明, 对每个词非监督估计其独立的参数集合会引入错误, 但是这个观点并不适用于常用词。因此 Kupiec 在等价类中没有包括 100 个最常用的词, 但是把它们当成独立的单个词类。

训练 初始化完成以后, 就可以用第 9 章中描述的前向 - 后向算法来训练隐马尔可夫模型。

标注 我们前面已经说过, VMM 标注和 HMM 标注之间的差别在于怎样训练模型, 而不是怎样进行标注。两种情况下训练后的正常结果都应该是一个隐马尔可夫模型。基于这个原因, 应用模型来标注是没有差别的。我们在使用隐马尔可夫模型进行标注与使用显马尔可夫模型进行标注时的 Viterbi 算法是完全一样的。

10.3.2 隐马尔可夫模型训练中的初始化的作用

一般使用对数似然准则作为前向 - 后向算法的终止条件(对数似然性不再增长的时候停止)。但是有些情况表明, 对于标注而言, 这个准则经常会导致训练过度。Elworthy(1994) 对这个问题做了详细的研究, 从表 10.6 中不同的条件开始训练 HMM。正如本章开头所述, D0 与 T0 的联合对应了显马尔可夫模型的训练。D1 对词汇概率进行了正确的排序(例如, 对于 make 而言, 标注 VB 的可能性比 NN 大), 但是概率的绝对值是随机的。D2 对于所有的词根据词性给出了同样的顺序。例如, 对于最常见的标注 t^j , $P(w|t^j)$ 比其他任何标注 t^k 的 $P(w|t^k)$ 都要大。D3 只保留了有关一个词可能的标记的信息, 顺序并不一定是正确的。T1 把转移概率初始化为大致相等的数字^①。

表 10.6 HMM 的参数初始化。D0, D1, D2 和 D3 是词汇初始化, T0 和 T1 是 Elworthy 研究得到的标记转移的初始化

D0	从带标记的训练语料库中得到的最大似然估计
D1	词汇概率的正确排序
D2	词汇概率和所有标记概率的比例
D3	一个词所有可能的标记赋予相同的词汇概率
T0	从带标记的训练语料库中得到的最大似然估计
T1	给所有转移相等概率

① 把完全相等的概率作为 EM 算法的初始条件通常是很糟糕的, 因为它们对应了通常很容易避免的不理想的局部最优优点。我们假设 D3 和 T1 表示带有轻微扰动并可以避免约束的大致相等的概率。

Elworthy(1994)发现了不同初始条件组合下的三种不同训练模式。在经典模式中,测试集上的性能随着每次训练迭代稳步提高。这种情况下对数似然准则对于停止训练是适用的,在早期极大值模式中,性能经历了多次迭代(经常是2次或者3次)提高以后,开始下降。在初始化极大值模式中,第一次迭代就开始导致性能下降。

应用HMM的比较典型的场合是可以得到词典,但是没有标注语料库作为训练数据的情况,即条件D3(或D2)以及T1。对于这种场合,训练遵循早期极大值模式。这就意味着我们在实践中必须小心,不要训练过度。一种处理方法是在每次迭代之后都在一个留存的验证集合上测试标注器,在性能下降的时候停止训练。

Elworthy也论证了Merialdo的发现,即当一个标注语料库(即使是中等规模)可以获得的时候也会导致前向-后向算法性能下降。也就是说,如果我们根据D0和T0进行初始化,就会得到初始化极大值模式。但是,一个有趣的变化是,如果训练和测试语料库相差很大,那么几次迭代确实能够提高性能(早期极大值模式)。这是在实践中经常出现的一种情况,因为我们经常必须面对没有相似的标注训练文本的情况。

总的来说,如果有足够大的训练文本,并且与程序所要处理的文本非常类似,那么我们将应用显马尔可夫模型。如果没有足够的训练文本,或者训练文本和测试文本差别很大,但是至少有一些词汇信息,我们就可以用前向-后向算法迭代几次。只有当根本就没有词汇信息时,我们才应当迭代很多次(10次或更多),但是在这种情况下我们并不能预期有好的性能。这种失败并不是前向-后向算法的一个缺陷,但是反映了前向-后向算法只是通过调节HMM的参数来使训练数据似然最大化。它用来减小交叉熵的变化也许并不符合我们真正的目标——得到指定了预定义标记集中的标记的词语。因此,在这个任务上它并不能实现优化。

10.4 基于转换的标注学习

在对马尔可夫模型的叙述中我们强调,马尔可夫假设对于自然语言语法结构的很多属性来说太粗糙了。由此产生的问题是,为什么我们不采用更加精巧的模型。我们可以把标注建立在先前的词语上(而不仅仅是前面的标记),或者可以通过采用四元或更高阶的语法标注器来利用比三元语法模型更多的上下文。

这种方法是不可行的,因为我们需要大量的参数。即使是三元语法模型,我们也不得不做一些平滑或插值工作,因为最大似然估计并不足够鲁棒。而这个问题会随着采用比现今马尔可夫模型更复杂的模型而加剧,特别是当我们希望把转移概率建立在词语条件上的时候。

现在来看看基于转换的标注。这种方法的一个优势在于可以利用更大范围的词汇和语法结构规则。特别地,标注可以建立在词语或更多的上下文上。通过选取和追加把一个初始的不完美标注器转化成一个只有很少错误的标注器,基于转换的标注编码了词语和标记之间复杂的依存关系。基于转换的标注器训练需要的决策量比估计大量的马尔可夫模型的参数要少一个级别。

基于转换的标注有两个重要组成部分:

- 允许的误差校正转换的详细说明
- 学习算法

我们需要一个已经标注好的语料库和一个词典作为输入数据。首先用最常用的标记来标注训练语料库中的每个词——这就是我们需要词典的原因。接下来学习算法构建了一个转换的排序表,它把初始的标注转化为接近正确的标注。通过再次初始化来选择每个词最常用的标记,并应用转换,这个排序表就可以用来标注新的文本。下面我们将对这些组成部分进行详细讨论。

10.4.1 转换

一个转换包含两个部分:一个触发环境和一条重写规则。重写规则的形式是 $t^1 \rightarrow t^2$, 表示“用标记 t^2 来替换 t^1 ”。Brill(1995a)制定了表 10.7 中的触发环境。这里的星号表示潜在重写的位置,方框表示寻找触发的位置。例如,第 5 行涉及到的触发环境为“标记 t^i 可以发生在前面三个位置之一。”

表 10.7 Brill 的基于转换的标注器中的触发环境。例如,第 5 行涉及到的触发环境为“标记 t^i 可以在前面三个位置中的一个发生。”第 9 行涉及到的触发环境为“标记 t^i 可以在前面的第二个位置出现,而标记 t^k 可以接着出现。”

方案	t_{i-3}	t_{i-2}	t_{i-1}	t_i	t_{i+1}	t_{i+2}	t_{i+3}
1				*			
2				*			
3				*			
4				*			
5				*			
6				*			
7				*			
8				*			
9				*			

给定触发环境所学到的转换类型的例子如表 10.8 所示。第一个转换指定名词在标记 TO 后面,应该被重新标记为动词。后面带有一些更专门触发器的转换会把一些词重新标回 NN (例如, *school in go to school*)。表 10.8 中的第二个转换应用在具有相同原形和过去式形式的动词上(例如 *cut* 和 *put*)。前面有一个情态动词使得动词不可能被用做过去式。第三种转换的一个例子是重新标注 *more valuable player* 中的 *more*。

表 10.8 在基于转换的标注中学习的一些转换的例子

源标记	目标标记	触发环境
NN	VB	前面的标记是 TO
VBP	VB	前面的三个标记之一是 MD
JJR	RBR	下一个标记是 JJ
VBP	VB	前面的两个词语之一是 $n't$

表 10.8 中的前三个转换是由标记触发的,第四个是由词来触发的。在 Penn 树库中,类似 *don't* 和 *shouldn't* 的词被划分成一个情态动词和 $n't$ 。类似第二个转换,这个也是把过去式变成了原形。前面的 $n't$ 使得后面更可能是一个原形而不是过去式。

词语触发环境可以建立在当前词或词语和词性标记的联合(当前词是 w^i ,而下一个标记是 t^j)上。

除了标记触发和词语触发转换之外,还有第三种转换形式。形态触发转换提供了一种很精致的方法来把未登录词处理整合到一般标注形式中。最初,未登录词如果大写则被标注为专有名词(NNP),否则被标注为普通名词(NN)。类似“如果未登录词的后缀是-s 那么用 NNS 替换 NN”的形态触发转换可以纠正错误。这些转换是通过采用与标注转换时一样的学习算法学习得到的。下面我们将描述学习算法。

10.4.2 学习算法

基于转换的标注学习算法选择了最佳的转换,并且确定了它们的应用次序,其工作方式如图 10.3 所示。

```

1  $C_0 :=$  corpus with each word tagged with its most frequent tag
3 for  $k := 0$  step 1 do
4    $v :=$  the transformation  $u_i$  that minimizes  $E(u_i(C_k))$ 
6   if  $(E(C_k) - E(v(C_k))) < \epsilon$  then break fi
7    $C_{k+1} := v(C_k)$ 
8    $\tau_{k+1} := v$ 
9 end
10 Output sequence:  $\tau_1, \dots, \tau_k$ 

```

图 10.3 基于转换的标注学习算法。 C_i 指语料库标注的第 i 次迭代, E 指错误率

最初我们用最常见的标记标注每个词。在每次迭代中,我们选择最可能减少错误率的转换(第 4 行),通过标注过的语料库 C_k 中被错误标注的词语的数目来衡量错误率 $E(C_k)$ 。当没有能够降低超过预先指定阈值 ϵ 大小的错误率的转换时将停止。这是一个转换最优序列的贪婪搜索过程。

对于怎样应用转换,即怎样精确地计算 $\tau_i(C_k)$,我们也有两个决策。首先,我们指定转换从左到右应用在输入上。其次,我们必须确定转换是否具有立即效果或延迟效果。在立即效果的情况下,同一个转换的应用会互相影响。Brill 实现了效果延迟转换,这个要简单一些,所谓延迟转换就是指“如果先前的标记是 A,那么 $A \rightarrow B$ ”将会把 AAAA 变成 ABBB。如果转换马上起作用,那么 AAAA 将会变成 ABAB。

这个标注模型的一个比较有趣的变化是用它来进行非监督学习,可作为 HMM 标注的一种可选方案。在 HMM 标注中,非监督学习惟一可以获得的信息是每个词有哪些标记是允许的。我们可以利用很多词只有一种词性标记这个事实,并把它作为选择转换的计分函数。例如,我们可以推导出 *The can is open* 中的 *can* 的正确标记是 NN(如果大多数没有歧义的词语在“AT_BEZ”环境下是标有这个标记的名词)。Brill(1995b)描述了一个基于这种思想的系统,并且获得了 95.6% 的标记准确率,对于非监督学习来说这是一个非常不错的结果。特别有意思的是,它没有训练过度问题,与我们上面所看到的 HMM 非常容易训练过度形成了鲜明的对比。这个问题稍后再讨论。

10.4.3 与其他模型的关系

决策树

基于转换的学习与决策树有一定的相似性(参见 16.1 节)。我们可以把决策树看成是一种标记所有树叶的机制,这些树叶都由一些带有其主要类别标记的节点来支配。我们向下遍历这棵树时,将重新标记一个子节点的树叶(如果它的标记与父节点不同)。这种查看决策树的方式表明了它与基于转换的学习的一些相似性。在基于转换的学习中我们也经历了一系列的重新标记,从而工作在越来越小的子数据集上。

原则上,Brill(1995a)指出,严格地讲基于转换的学习比决策树学习更强大。也就是说,有一些分类任务能够被基于转换的学习解决,使用决策树却解决不了。但是基于转换的学习的这种“额外力量”在 NLP 的应用中并不是很清楚。

这两种方法的主要的实际区别是,在决策树中训练数据在每个节点上都被划分,而且我们对每个节点都应用了一个不同的转换序列(这个序列对应于从根节点到那个节点的决策路径)。在基于转换的学习中,在学习得到的转换表中的每个转换都会应用在所有的数据上(在触发环境符合时会引发重写)。由此,我们可以直接最小化我们感兴趣的指标(例如,在进行标注时的标注错误率),这不同于 HMM 和决策树中所使用的类似熵的间接度量。如果我们直接最小化了决策树学习中的错误率,就很容易在每个叶子节点上达到 100% 的准确率。但是,在新数据上的性能却很差,因为每个叶子节点都是在训练数据的任意属性没有一般化的基础上形成的。基于转换的学习看上去对于这种形式的过适应(Ramshaw and Marcus 1994)有令人惊奇的免疫力。这可以由我们总是在整个数据集上学习的事实来部分解释。

为了拥有鲁棒性,我们需要付出的一个代价是需要搜索很大的转换空间,因此一个基于转换的学习的简单实现的效率非常低。但是,通常可以使用一些更加智能和有效的方法来搜索空间(Brill 1995a)。

一般的概率模型

与概率模型(包括决策树)相比,基于转换的学习并不能使用概率思想所提供的一些标准方法。例如,在 k -best 标注的概率模型中(一个把大量的标注概率假设传递到下游模块,例如句法分析器的标注模块),额外的工作是不必要的。

有可能通过“把标记 A 加入 B,如果…”形式的规则把基于转换的标注扩展成 k -best 标注器,因此有些词可能会被标上多个标记。但是,我们无法评估每个标记的可能性有多大。在某些情况下第一个标记的可能性是下一个最好的标记的可能性的 100 倍,而在其他的情况下所有的标记都是等可能的。这类知识对于构造一个句法分析器很关键。

学习方法的一个重要特征是先验知识编码的方式。基于转换的标注和概率方法有不同的能力。最合适的触发环境的模板详细说明提供了一个有力的途径,使得在基于转换的学习中学习器趋向于好的一般化。表 10.7 中的模板看上去很好理解,但这只是因为我们知道语法结构规则,还有其他明显不合适的大量模板(例如,“句子中前面的偶数位置是名词”)。

与之相对应的是,概率马尔可夫模型使得每个词的不同标记的先验似然性能够更容易地

被精确编码(例如,最可能的标记 10 倍于或者只是 1.5 倍于第二可能的标记)。对于基于转换的学习的标注中的学习器,我们能给予的知识只有哪个标记是最可能的。

10.4.4 自动机

读者也许想知道为什么本书中描述了基于转换的标注,因为我们说过不会包括面向规则的方法。虽然基于转换的标注有一个规则部分,但是它也有一个定量部分。我们在基于语料库或语料库计数的定量方法的意义上稍微放宽了对统计 NLP 的限制,而并不仅仅只是应用了概率思想框架。基于转换的标注在这个意义上很明显是统计 NLP 方法,因为转换是基于量化规则选择出来的。

但是,转换的量化评估(通过它们提高了多少错误率)只出现在训练中。一旦学习完成,基于转换的标注就完全是符号化的了。这意味着基于转换的标注可以被转化为其他在标注性能上等效的符号方法,而具有如时间效率的一些其他优点。

下面是 Roche and Schabes(1995)采用的方法。把一个基于转换的标注器转化成一个等效的有限状态转换器,这个有限状态自动机的每条弧上有一对符号、一个输入符和一个输出符(有些情况下在经过一条弧时可能有几个输出符)。一个有限状态转换器通过一个输入字符串时,消耗它所经过的弧的输入符,并在同样的弧上发射输出符,把输入字符串转化为输出字符串。

Roche 和 Schabes 提出的构造算法有第四步。首先,每个转换都被转化为一个有限状态转换器。第二步,转换器被转化为它的局部扩展。简单地说,转换器 f_1 的局部扩展 f_2 被构造出来,使得 f_2 在整个输入串上运行一遍与 f_1 在输入串上的每个位置运行一遍的效果是一样的。这一步需要注意类似下面的这种情况。假设我们有一个转换器实现了转换“如果前面的两个符号里有一个是 C,那么用 B 替换 A”,这个转换器将会有一条弧,它的输入符号为 A,输出符号为 B。因此,对于一个类似 CAA 的输入串,我们不得不运行两次(在第二个和第三个位置),以保证正确地把 CAA 转换为 CBB。但是,构建了一个局部扩展以后,一遍就可以实现转换。

第三步,我们把所有的转换器组合成一个转换器,它的效果与在序列中运行单个转换器是一样的。单个转换器通常是不确定的。任何时候转换器的存储中都要保存一个事件(类似“C 在位置 i 出现”),为了实现这一点需要开辟两条路径,一条假设受前面的 C 影响的标记会在后面出现,另一条假设这类标记不会出现。合适的路径会走得更远,不合适的路径将会在字符串的某个合适位置被丢弃。但是,这种不确定性是没有效率的,因此第四步便是把不确定的转换器转化成确定的。通常这是不可能的,因为不确定的转换器会在存储器中保存一个任意长度的序列的事件,而确定的转换器是做不到这一点的。但是,Roche 和 Schabes 表明基于转换的标注中用到的转换器并没有这种属性,因此我们总是可以把一个基于转换的标注器转化成一个确定的有限状态转换器。

确定的有限状态转换器的最大优点在于速度快。一个基于转换的标注器可以用 RKn 个基本步骤来标注一个文本,其中 R 是转换的个数, K 是触发环境的长度, n 是输入文本的长度(Roche and Schabes 1995:231)。与之对比的是,有限状态转换器对于输入文本的长度是线性的,并且只有一个很小的常数。基本上,当我们读一个词的时候,只是从一种状态跳到下一个状态,查找它最可能的标记(初始状态)并输出正确的标记。这使得它可能达到每秒处理上万个词的速度。马尔可夫模型标注器的速度低得多。这就意味着基于转换器的标注在操作中只加入了一个很小的开支,相当于从磁盘上读入输入文本。并且,它需要的时间相比于类似句法

分析或信息理解的后续处理步骤可以忽略不计。

也有一些将隐马尔可夫模型转化成有限状态转换器的工作(Kempe 1997)。但是,在这种情况下,我们并不能获得完全等效的模型,因为自动机不能完美地模仿 Viterbi 算法中需要的浮点运算。

10.4.5 小结

基于转换的标注算法的最大优点在于它能够把标注决策建立在一个比我们前面看到的概率模型更加丰富的事件集合上。例如,从左到右的信息能够同时被使用,而且单独的词(不仅仅是它们的标注)可以影响邻近词的标注。基于转换的标注可以适应更大的触发环境集合的一个原因可能是它主要处理了比概率简单的二元信息。

同时,有研究表明转换比概率标注中的转移和词语生成更容易理解和修改。但是,要预见改变序列中的一个转换的效果是很难的,因为当几十个转换被应用到序列中并且每个都依赖于前一个输出的时候,将会出现复杂的相互影响。

基于转换的标注的理论基础方面的工作目前仍在进行。例如,基于转换的学习不会出现过适应的事实,仍然只是一个还没有被很好理解的经验结果。

学习和标注在基于转换的标注中都是很简单和直观的。无论这种简单性是否可以作为在基于转换的标注器和概率标注器之间进行选择的基本原则,概率模型在处理不确定和确定类别的先验知识时的能力是否更重要,这些都需要建立在很多因素上,比如标注器是什么系统的一部分,以及这些系统上的工作是否更适合基于规则或概率方法。

除了标注,基于转换的学习还被应用于句法分析(Brill 1993b)、介词短语附着(Brill and Resnik 1994)以及语义消歧(Dini et al. 1998)上。

10.5 其他模型和语言

10.5.1 其他标注方法

在过去的10年中,标注在NLP中是最活跃的研究领域之一。这里我们只能谈到最重要的三种方法。其他很多概率和定量的方法都被用在标注上,包括在第16章中将涉及的所有方法:神经元网络(Beneito et al. 1989)、决策树(Schmid 1994)、基于存储的学习(或 k 近邻方法)(Daelemans et al. 1996)以及最大熵模型(Ratnaparkhi 1996)^①。

也有一些人研究怎样使用最少的人力来构建标注语料库(Brill et al. 1990)。当一个没有标注好的训练语料库的语言需要处理时,或者当已经有一个标注好的训练语料库但遇到的文本非常不同,使得现有的标注语料库没有用的时候,便会出现这些问题。

最后,一些研究人员已经尝试了一些方法来自动构建一个标记集,目的是创造对一种语言或特殊类型的文本适用的语法结构类别(Schütze 1995; McMahon and Smith 1996)。

10.5.2 其他非英语语言

我们这里只涉及到英语的词性标注。我们发现,英语是一种能够尝试在一个词语序列中

^① Ratnaparkhi's 的标注器可以公开获得,它是当前性能最好的统计标注器之一。参见本书配套网站。

从词语的位置来推导它的语法类别的语言。在许多其他语言中,词语顺序是很自由的,而且周围的词语对于词性也只提供了很少的信息。但是,在大多数语言中,词语丰富的变形比英语提供了更多的词性信息。把标注器作为高层次的多语言 NLP 任务中的有用预处理器,以此来对标注器进行充分评估,这只能出现在可获得来自大范围语言的足够实验结果之后。

尽管存在这些限制,现在还是已经有了很多对于标注的研究,至少对于欧洲语言是这样的。这些研究表明,其他语言的标注准确率与英语是可比较的(Dermatas and Kokkinakis 1995; Kempe 1997),然而由于标记集的不可比性(标记集不是普遍适用的,但是都对单独语言的特殊功能类别进行了编码)导致了这种比较很难实现。

10.6 标注准确率和标注器的应用

10.6.1 标注准确率

现今报道的标注的准确率数字大多数都在 95% ~ 97%,这是建立在对所有的词进行计算的基础上的。一些作者只给出了带有歧义的词语的准确率,在这种情况下准确率当然很低。但是,标注性能主要依赖于下面几个因素:

- **可以获得的训练数据量** 通常越多越好。
- **标记集** 通常,标记集越大,潜在歧义就会越多,标注任务就越困难(参见 4.3.2 节中的讨论)。例如,一些标记集中区分了介词 *to* 和不定式标志 *to*,有的却没有。使用后一个标记集就不会把 *to* 标注错误。
- **训练语料库及词典与应用语料库之间的差别** 如果训练文本和应用文本来自同一个语料源(例如,同一时期的某个报纸),那么准确率就会很高。通常,研究论文中展示的结果都来自这种情况。如果应用文本来自于不同的时期,或者来自于不同的语源,甚至来自与训练文本不同类型的文本(例如,科学文本与新闻文本对比),那么性能就会很差^①。
- **未登录词** 最后一点特殊情况是词典的覆盖率。大量未登录词的出现会大大降低标注算法的性能。在标注一些专门技术领域中的材料时没有出现在词典中的词的百分比会很高。

上面四个条件中的任何一个发生改变都会影响标注准确率,有时候影响会很大。如果训练集很小,标记集很大,测试语料库与训练语料库差别很大,或者我们要面对比预期多得多的未登录词,那么性能会比上面引用的数字低得多。有必要强调一下,这些外在条件比标注方法的选择对于性能有更大的影响,特别是当报告的方法差别在半个百分点的级别上的时候。

当我们评价一个‘dumb’标注器的相对较高的准确率的时候,需要考虑外部因素的影响,这些标注器总是选择词语最常用的标记。这种标注器在较好的条件下可以获得 90% 左右的准确率(Charniak et al. 1993)。如果我们认识到它使用的词典(Charniak et al. 1993)基于应用语料库(Brown语料库),那么这么高的准确率并不令人感到惊奇。大量手工努力获得的资源使得现在很容易确定Brown语料库中的一个词的最常用标记是什么。因此,利用了词典信息的标注器取得了很好的性能并不令人感到惊奇。自动标注器最开始用来预处理Brown语料库时只获得了 77% 的准确率(Greene and Rubin 1971),其部分原因是它的非概率本质,但大部分原因是它不能依赖于—

^① 参见 Elwoythy(1994)以及 Samuelsson and Voutilainen(1997)中考察训练集不同程度相似性对标注性能影响的实验。

个适合于语料库应用并给定词语所使用的不同词性频率的大词典。

即使在我们有一个很好的词典并且最经常使用工作得很好的标注策略的情况下,一个标注器在 90% ~ 100% 的准确率范围之间到底做得好到什么程度仍然很重要。例如,一个准确率为 97% 的标注器有 63% 的可能性使得一个 15 个词的句子得到完全正确的标注,而准确率为 98% 的标注器则可以达到有 74% 的可能性。因此,即使是很小的提高也会造成在应用上的很大差别。

有一个性能最好的非定量标注形式,即 EngCG(英语受限语法),由 Helsinki 大学开发。Samuelsson and Voutilainen(1997)表明它的性能比马尔可夫模型标注器更好,特别是在训练和测试语料不同源的情况下^①。在 EngCG 中,手写规则被编译成有限状态自动机(Karlsson et al. 1995; Voutilainen 1995)。其基本思想有点类似于基于转换的学习,除了人工(而不是用算法)迭代地修改标注规则集合以最小化错误率以外。在每次迭代中,将当前规则集运行在语料库上,并尝试修改规则,使得最严重的错误能够得到正确的处理。这种思想等同于写了一个小的标注专家系统。据称非常熟悉这种方法的人写出这种标注器并不比构造一个 HMM 标注器更费力(Chanod and Tapanainen 1995),虽然有争论说 HMM 标注方法更容易理解。

最后,我们给出一些常见错误的例子。表 10.9 展示了由 Kupiec(1992b)报告的一些常见的错误类型的例子。例子中的短语和片段都是有歧义的,说明有必要比马尔可夫模型涉及到更多的语义上下文或者结构上下文。在语法结构上,词语 *executive* 可以是一个形容词,也可以是一个名词。短语 *more important issue* 可以指大量的重要问题,或者问题更重要。词语 *up* 在 *running up a hill* 中用做一个介词,在 *running up a bill* 中用做一个小品词。最后,依靠内嵌, *needed* 在下面的两个来自于(Kupiec 1992b)的句子中可以是过去分词或者过去式:

a. The loan needed to meet rising costs of health care.

b. They cannot now handle the loan needed to meet rising costs of health (10.12) care.

表 10.9 概率标注器中常见的错误例子

正确标记	标注错误	例子
noun singular	adjective	<i>an <u>executive</u> order</i>
adjective	adverb	<i><u>more</u> important issues</i>
preposition	particle	<i>He ran <u>up</u> a big ...</i>
past tense	past participle	<i>loan <u>needed</u> to meet</i>
past participle	past tense	<i>loan <u>needed</u> to meet</i>

表 10.10 显示了(Franz 1995)中标注器的混乱矩阵的一部分。每一行显示了某个类别的词语被标注器给予不同标记的次数百分比。从某种意义上说,这个结果并不令人惊奇。错误发生在兼类的情况下。但是,特别值得注意的是标注小品词的低准确率,因为所有的小品词同时也可以当成介词来用。小品词和介词之间的差别很真实,但又很微妙,实际上即使是在手工标注的语料库中也不一定能准确地将其区分开^②。

① 该论文中报告的 EngCG 的准确率数字超过 99%,而 HMM 模型标注器的准确率数字超过 95%,但是对二者进行比较是很困难的,因为 EngCG 还没有解决一些歧义。在某些情况下 EngCG 会返回不止一个标注。

② 因此,我们将会在第 12 章中看到小品词在概率句法分析器的评估中经常被忽略。

表 10.10 词性标注混乱矩阵的一部分。对于每个标记,表中每一行表明标注器把该类别的符号分配给不同标记的次数百分比(因此,在整个混乱矩阵中,每一行的百分比之和为100%,但是这里出现的只是表的一部分)。该表基于(Franz1995)

正确的 标记	标注器标注的标记							
	DT	IN	JJ	NN	RB	RP	VB	VBG
DT	99.4	0.3			0.3			
IN	0.4	97.5			1.5	0.5		
JJ		0.1	93.9	1.8	0.9		0.1	0.4
NN			2.2	95.5			0.2	0.4
RB	0.2	2.4	2.2	0.6	93.2	1.2		
RP		24.7		1.1	12.6	61.5		
VB			0.3	1.4			96.0	
VBG			2.5	4.4				93.0

10.6.2 标注的应用

许多 NLP 的应用将会从没有语法结构歧义的文本中获益,当前对于标注的广泛兴趣便是建立在此信仰之上的。很奇怪会有更多的论文独立地研究标注,甚至超过了把标注应用在当前感兴趣的任务中。这里我们总结一下用到标注器的一些最重要的应用。

大多数应用在标注之后还需要一个额外的处理步骤:部分句法分析。部分句法分析会涉及到语法结构分析细节的不同层次。最简单的部分句法分析器限制在找到一个句子的名词短语上。更加精巧的方法则为名词短语分配了语法功能(主语、直接宾语和间接宾语),并且给出了附着的部分信息,例如,“这个名词短语附着在另一个短语(没有指定)的右边”。

有一种利用马尔可夫模型的很精妙的方法可以用来实现名词短语识别,参见 Church (1988),在(Abney 1996a)中则会找到更好的描述。我们可以得到标注器的输出并形成二元标记序列。例如,NN VBZ RB AT NN 可以转换成 NN-VBZ VBZ-RB RB-AT AT-NN。接下来这个二元标记序列用 5 个符号来进行标注:名词短语开始、名词短语结尾、名词短语内部、名词短语外部(即,这个二元标记并不是名词短语的一部分)以及名词短语之间(即,这个二元标记的左边和右边都紧接着一个名词短语)。于是,名词短语就是位于名词短语开始符号(或名词短语之间符号)和名词短语结束符号(或名词短语之间符号)之间的所有标记序列,其间有名词短语内部符号。

最著名的部分句法分析方法是 Fidditch,是由 Hindle(1994)在 20 世纪 80 年代初开发的,另外还有由 Abney(1991)提出的称为“语块句法分析”的方法。这两个系统并没有使用标注器,因为它们出现在可以广泛获得标注器之前。也可参见(Grefenstette 1994)。比当前部分句法分析器更强大并且试图跨过浅层和完全句法分析之间的断层的两个方法是 XTAG 系统(Doran et al. 1994)和语块标注系统(Brants and Skut 1998; Skut and Brants 1998)。

在很多建立在标注器上的部分句法分析器系统中,部分句法分析通过匹配标注器的正则表达式来完成。例如,一个简单名词短语可以定义为一个由冠词(AT)、任意数量的形容词(JJ)和一个名词(NN)组成的序列。这对应正则表达式 $AT \cdot JJ^* \cdot NN$ 。因为这些系统集中在最终应用上,而不是在部分句法分析上,所以我们并不把它们归结到“部分句法分析”的题目下面,而是根据它们趋向的应用进行分类。部分句法分析(以及标注)的一个比较好的介绍参见(Abney 1996a)。

在与部分句法分析的结合中,标注的另一个重要应用是词汇获取。我们已在第8章中讲述过。

另一个重要的应用是信息抽取(也可以被称为信息理解、数据抽取或文本数据挖掘)。信息抽取的目标是找到模板中预定义槽中的值。例如,天气预报的一个模板有天气状况(飓风或暴风雪)、事件位置(the San Francisco Bay Area)、事件的时间(星期日,1998年1月11日)以及事件效果(能量损耗、交通事故等)之类的槽。标注和部分句法分析帮助识别了作为槽填充者的实体及其之间的关系。最近的关于信息抽取的概述文章是(Cardie 1997)。在某种程度上我们可以把信息抽取当成标注,只是标注的是语义类别而不是语法词性。但是,在实践中采用了非常不同的技术,因为局部序列比语法类别给出了更少的语义类别信息。

标注和部分句法分析也能够用来在信息检索中找到好的索引词。匹配用户要求和文档的最好单元通常不是单个词。类似于 *United States of America* 和 *secondary education* 的短语如果被打散成词,会丢失很多含义。如果标注和部分句法分析被用在名词短语识别上,并且查询-文本匹配建立在比单个词更有意义的单元上,信息检索的性能将会得到很大提高(Fagan 1987; Smeaton 1992; Strzalkowski 1995)。一个相关的研究领域是短语规范化,其中短语变形被规范化,并且用同样的基本单元表示(例如, *book publishing* 和 *publishing of books*)。参见(Jacquemin et al. 1997)。

最后,还有一些称为问题回答系统的工作,通过返回合适的名词短语(例如地点、人物或时间),试图回答用户格式化的问题(Kupiec 1993b; Burke et al. 1997)。例如,问题 *Who killed President Kennedy?* 会用一个名词短语 *Oswald* 来回答,而不是像常见的信息检索系统那样返回一个文档列表。再者,为了确定用户查找的是哪一类实体或者为了确定它与问题中涉及的其他名词短语的关系,查询分析也需要进行标注或者部分分析。

我们用一个负面结果来做总结:目前最好的词汇化概率句法分析器并不用标注器作为预处理器(Charniak 1997a),它们从未标注的文本开始自己进行标注会工作得更好。因此,标注器看起来是一个不重要的部分,它可以给很多应用任务提供充足的信息,却不是所有应用任务中的一个必要的预处理过程。

10.7 深入阅读

20世纪60年代初期,使用马尔可夫模型进行自然语言建模的早期工作的停滞不前,部分是因为Chomsky对于马尔可夫模型不充分性的批评(Chomsky 1957: Ch.3),缺少训练数据和计算资源也起了一定的作用。Chomsky的批评仍然在起作用:马尔可夫链并不能完全模型化自然语言,特别是它们不能描述递归结构,例外情况参见Ristad and Thomas (1997)。

最早的标注器是一些从词典中查找词语类别的简单程序。试图基于语段上下文分配标记的最早的著名程序是出现在(Klein and Simmons 1963)中的基于规则的程序,与(Salton and Thorpe 1962)中的想法大致相同。Klein和Simmons使用了术语‘标记’和‘标注’,很明显相当于‘码’和‘编码’。我们知道的最早的概率标注器是(Stolz et al. 1965)。这个程序最初通过使用词典、变形规则以及其他一些特殊规则来为一些词(包括一些功能词)分配标记。剩下的开放词类使用从标记序列中计算得到的条件概率来进行标注。毫无疑问,这是一个不完备的概率模型。

我们应该把荣誉给予两个团体,一个是 Brown 大学,一个是 Lancaster 大学,两者花费了巨大的资源来标注了两个大的语料库, Brown 语料库和 Lancaster-Oslo-Bergen (LOB) 语料库。两个团体认识到带有标注信息的语料库对于语料库的深入研究是无价的。没有这两个语料库,词性标注方面的进步即使不是不可能也是很困难的。可以获得大量标注数据毫无疑问是标注成为这样一个活跃研究领域的重要原因。

Brown 语料库采用了一个基于规则的标注器 TAGGIT (Greene and Rubin 1971) 自动进行了预标注。这个标注器使用词汇信息只是为了限制词语标记,并且只在词语周围的上下文没有歧义标注的情况下应用标注规则。标注之后利用手工又进行了校正,这项工作花费了很多年的努力,并且之后为了进行大量的定量工作又补充了训练数据。

第一个马尔可夫模型标注器是作为 LOB 标注工作的一部分由 Lancaster 大学构建的,参见 (Garside et al. 1987; Marshall 1987)。这个标注器的核心是使用二元标记序列概率,同时有限地使用高阶上下文,但是把一个词分配给不同词性的不同概率是由特别的折扣因素来处理的。同时,基于词语概率和转移概率标注的马尔可夫标注器由 Church (1988) 和 DeRose (1988) 引入。

在 NLP 中定量方法出现的开始时期, NLP 领域内概率思想的知识水平是很低的,以至于早期论文中的一个常见错误是在马尔可夫模型中计算下一个标记的概率时使用式 (10.14) 而不是式 (10.13)。虽然式 (10.14) 好像更加直观。毕竟,我们是在查看一个词并且确定它的标记,因此假设词语是给定的而标记建立在词语之上,这并不是完全没有道理的。

$$\arg \max_{t_{1,n}} P(t_{1,n} | w_{1,n}) = \prod_{i=1}^n [P(w_i | t_i) \times P(t_i | t_{i-1})] \quad (10.13)$$

$$\arg \max_{t_{1,n}} P(t_{1,n} | w_{1,n}) = \prod_{i=1}^n [P(t_i | w_i) \times P(t_i | t_{i-1})] \quad (10.14)$$

但是,实际上式 (10.14) 是不正确的,使用它会导致很低的性能 (Charniak et al. 1993)。

尽管 Church 和 DeRose 的工作对于计算语言学领域统计方法的兴起很关键,但是隐马尔可夫模型标注方面的工作实际上最早开始于纽约和巴黎的 IBM 研究中心。Jelinek (1985) 以及 Derouault and Merialdo (1986) 被广泛地引用。早期的参考文献是 Bahl and Mercer (1976) 以及 Baker (1975), 对 Elaine Rich 的工作有很大贡献。其他的概率标注的早期工作包括 (Eeg-Olofsson 1985; Foster 1991)。

名词到动词 (来自名词的动词) 的转化以及它的多产性 (productivity) 的语言学讨论可以在 Clark and Clark (1979) 中找到。Huddleston (1984: ch. 3) 很好地讨论了词性的传统定义, 它们的缺陷以及在现代结构主义语言学中使用的词类或词性的概念。

10.8 习题

10.1 [★] 还有其他哪些不能被马尔可夫链正确模写的语言学现象? 这些现象所共有的一般语言性质是什么?

10.2 [★] 为什么语言建模中时间不变性是有问题的?

- 10.3 [★]式(10.4)和式(10.5)都是一些简化的假设。请读者分别给出两个词语独立性(见式(10.4)),一个词和其前后标记的独立性(见式(10.5))不成立的例子。
- 10.4 [★]给出表 10.3 中的数据,计算图 10.1 中 $P(AT|PERIOD)$, $P(NN|AT)$, $P(BEZ|NN)$, $P(IN|BEZ)$, $P(AT|IN)$ 以及 $P(PERIOD|NN)$ 的最大似然估计。假设标记总的出现次数可以通过对每行的数字求和得到(例如,对于 BEZ 是 $1973 + 426 + 187$)。
- 10.5 [★]给定表 10.4 中的数据,计算图 10.1 中 $P(bear|t^k)$, $P(is|t^k)$, $P(move|t^k)$, $P(president|t^k)$, $P(progress|t^k)$ 以及 $P(the|t^k)$ 的最大似然估计。从表 10.3 中取得标记出现的总次数。
- 10.6 [★]计算下列概率:
- $P(AT\ NN\ BEZ\ IN\ AT\ NN | \text{The bear is on the move.})$
 - $P(AT\ NN\ BEZ\ IN\ AT\ VB | \text{The bear is on the move.})$
- 10.7 [★]请读者基于前面习题中的概率估计,使用 Viterbi 算法标注下面的句子。
- The bear is on the move. (10.8)
- 10.8 [★]一些标注序列概率的大数据集以及推荐的习题都可以从网站上获得。
- 10.9 [★]给定表 10.5 中的(构造)数据以及 Weischedel et al.'s 的未登录词模型,计算 $P(fenestration|t^k)$, $P(fenestrates|t^k)$, $P(palladio|t^k)$, $P(palladios|t^k)$, $P(Palladio|t^k)$, $P(Palladios|t^k)$ 以及 $P(guessimating|t^k)$ 。假设 NNP, NN, NNS, VBG 以及 VBZ 是可能的标注词性。这些估计是正确的吗? 还有其他的特征可以用来获得更好的结果吗?
- 10.10 [★★]利用从网站上得到的数据重新计算表 10.5,以得到更好的概率估计。
- 10.11 [★]请读者用加 1 法重新计算习题 10.4 和习题 10.5 中的概率估计。
- 10.12 [★]在前面引入 HMM 标注模型的时候,我们说过随机地初始化模型参数(没有词典信息)并不是 EM 算法的一个有效起始点。为什么呢? 如果我们只有下面的 8 种标记并且随机地初始化 HMM,会出现什么情况呢? 8 种标记为:介词、动词、副词、形容词、名词、冠词、连词以及助词。提示:EM 算法会集中在对于对数似然性(数量最大化)有最大影响的高频事件上。
这种初始化与 D3 有什么区别?
- 10.13 [★]EM 算法在每次迭代中可以提高给定了数据的模型的对数似然性。Elworthy 和 Merialdo 的实验结果表明,标注准确率通常会随着进一步训练而降低,他们的结果是一致的吗?
- 10.14 [★]在 Jelinek 和 Kupiec 的参数初始化方法中,得到先验知识的关键点是哪些词语生成概率是 0 而哪些不是。这里隐含的假设是,如果一个生成概率开始就被设成 0,那么会在训练中保持为 0。试说明这就是在第 9 章中涉及到前向-后向算法时的情况。
- 10.15 [★★]下载得到 Xerox 标注器(参见网站上的指示)并标注从网站上得到的文本。

-
- 10.16 [★]基于转换的学习是贪婪搜索的一种形式。贪婪搜索能够期望找到最优的转换序列吗?还有什么其他的办法?
- 10.17 [★]Brill(1995a)中的大多数触发环境都涉及到前文。为什么?除了英语以外你认为还有其他语言有这种情况吗?
- 10.18 [★]词语和标记的可能触发环境集合在(Brill 1995a)中是不同的。例如,“前面的三个标记中某一个 X”在触发环境中是允许的,但是“前面的三个词语中有一个是 X”却是不允许的。是什么原因造成了这种不同?考虑词语和标记不同的搜索空间大小。
- 10.19 [★★]除了选择最常用的标记进行初始化以外,我们也可以为所有的词分配一个同样的标记(比如 NN),或者使用另一个标注器的输出作为初始标记,然后用基于转换的标注器提高性能。讨论不同初始化的相对优点。
- 10.20 [★]下载得到 Brill 标注器(参见网站上的指示),并标注从网站上得到的文本。
- 10.21 [★]通常,我们所要标注的文本并没有被分割成句子。第4章中介绍了一种确认句子边界的算法,同时还有一个语料库处理本质的一般性介绍。
我们能够把句子边界检测整合到本章中介绍的标注方法中吗?需要改变些什么呢?你期望通过标注的手段使句子边界检测能够取得什么样的效果呢?
- 10.22 [★★]下载得到 MULTEXT 标注器(参见网站上的指示),并标注从网站上得到的一些非英语文本。

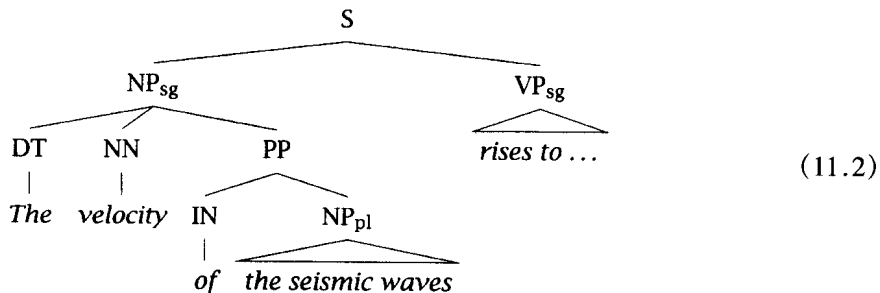
第 11 章 概率上下文无关文法

人们说话的方式(即使是人醉酒之后的胡言乱语)都存在一些结构和规则。语言学中的语法分析的目标就是努力分离出这些语法结构。现在我们一般所说的语法实际上就是指描述词语排列的方法,而排列的方式或者直接基于词,或者基于词类。在这一章中,我们暂且抛开这些线性的 n 元语法模型或 HMM 标注模型,探讨一些更复杂的语法概念。

即使在最传统的语法形式中,语法也不仅仅意味着线性词语之间的简单顺序。它必须描述词语是如何聚集的,以及中心词和附属词之间是如何互相联系的这一类问题。在最近大约 50 年之间,解决这个问题主要方法是为每个句子构造一个树结构,正如在第 3 章中所看到的。语言有着复杂的递归结构,这些基于树的模型(与马尔可夫模型不同)可以体现这一点。例如 Kupiec(1992b)提出,基于 HMM 的词性标注器在处理如下结构的时候会出现问题:

The velocity of the seismic waves rises to ... (11.1)

其原因是,在一般情况下一个复数名词之后不会跟一个单数动词(这里是 *rises*)。Kupiec 的解决办法是扩展这个 HMM 模型,以便它能够识别出一定数量的 NP 结构,这些 NP 结构在模型中作为高阶的上下文扩展粘附在基本的一阶 HMM 模型上。我们暂且把这个解决方案的技术细节放到一边。观察分析树(11.2)中的句法结构树,我们发现主谓数一致的性质实际反映了句子的层次结构,而不是词语的线性顺序。



动词在数上和它之前的名词短语的中心词 *velocity* 一致,而不是和它之前的名词的数一致。

最简单的表达递归嵌入概念的概率模型是 PCFG, 概率(有时候也称为随机)上下文无关文法就是一个为规则增添了概率的简单 CFG, 指明了不同重写规则的可能性大小。PCFG 的优点很多, 这也是本章我们详细讨论它的原因所在。PCFG 对于树结构来说是最简单、最自然的概率模型, 其数学背景很容易理解, 其算法是 HMM 所使用算法的一个自然发展, 而且 PCFG 提供了一个方便的计算工具, 能够模拟各种形式的概率条件(正如我们将在 12.1.9 节中所描述的)。不过, PCFG 只是构造句法结构概率模型的众多方法中的一种, 在下一章中我们将在更广泛的领域内研究概率句法分析。

一个 PCFG G 包括:

- 一个终结符集合, $\{w_k\}, k = 1, \dots, V$

- 一个非终结符集, $\{N^i\}, i = 1, \dots, n$
- 一个指定的初始符, N^1
- 一个规则集合, $\{N^i \rightarrow \zeta^j\}$ (在这里 ζ^j 是指一个终结符或非终结符序列)
- 一个对应的规则概率集合如下:

$$\forall i \sum_j P(N^i \rightarrow \zeta^j) = 1 \quad (11.3)$$

有一点读者需要注意, 即 $P(N^i \rightarrow \zeta^j)$ 仅仅是缩写形式, 完整的表达式应该是 $P(N^i \rightarrow \zeta^j | N^i)$ 。即, 我们给出了某个特定中心词的子女的概率分布形式, 这种语法可以用于分析和生成语言的句子, 今后我们会对这两个表达式的使用不做区分。

在使用 PCFG 分析一个句子之前, 我们需要建立一些符号。我们把一些待分析的句子表示成一个词序列 $w_1 \dots w_m$, w_{ab} 表示子序列 $w_a \dots w_b$ 。单箭头 $\rightarrow$ 表示语法的一个单独重写操作。作为一个或者多个重写规则的结果, 非终结符 N^j 表示成一个词语序列 $w_a \dots w_b$, 于是我们可以说 N^j 支配着词语 $w_a \dots w_b$, 写为 $N^j \Rightarrow w_a \dots w_b$ 或 $\text{yield}(N^j) = w_a \dots w_b$ 。这种情况在分析树 (11.4) 中给出了描述: 一个以非终结符 N^j 为根的子树支配了词串 $w_a \dots w_b$ 中的所有词, 如下所示:

$$\begin{array}{c} N^j \\ \wedge \\ w_a \dots w_b \end{array} \quad (11.4)$$

当我们说 N_j 是跨越词串中位置 a 和 b 的一个非终结符时, 并没有指出这个序列实际所包含的词, 我们写成 N_{ab}^j 。这些符号在表 11.1 中给出了解释。

表 11.1 PCFG 这一章中的符号

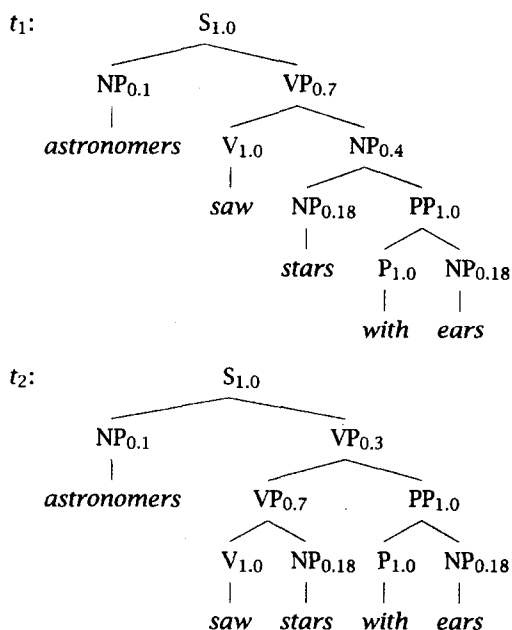
符号	含义
G	语法(PCFG)
$\mathcal{L}$	语言(由一个语法生成的或者被一个语法接受的)
t	分析树
$\{N^1, \dots, N^n\}$	非终结符表(N^1 是初始符)
$\{W^1, \dots, W^v\}$	终结符表
$w_1, \dots, w_m$	待分析的句子
N_{pq}^j	位置 p 到 q 的词串形成的非终结符 N^j
$\alpha_j(p, q)$	外部概率(11.15)
$\beta_j(p, q)$	内部概率(11.14)

一个句子的概率(根据语法 G)由以下公式给出:

$$\begin{aligned} P(w_{1m}) &= \sum_t P(w_{1m}, t) \quad t \text{ 是句子的分析树} \\ &= \sum_{\{t: \text{yield}(t) = w_{1m}\}} P(t) \end{aligned} \quad (11.5)$$

而且, 我们很容易利用 PCFG 模型确定一棵树的概率, 只需把构造它的各个子树的规则概率相乘即可。

例 1 假定有表 11.2 中的语法, 句子 *astronomers saw stars with ears* 则有两个概率句法分析树, 如图 11.1 所示。



$$P(t_1) = 1.0 \times 0.1 \times 0.7 \times 1.0 \times 0.4 \times 0.18 \times 1.0 \times 1.0 \times 0.18 \\ = 0.000\ 907\ 2$$

$$P(t_2) = 1.0 \times 0.1 \times 0.3 \times 0.7 \times 1.0 \times 0.18 \times 1.0 \times 1.0 \times 0.18 \\ = 0.000\ 680\ 4$$

$$P(w_{15}) = P(t_1) + P(t_2) = 0.001\ 587\ 6$$

图 11.1 两个分析树, 其概率以及句子概率。对应的句子是 *astronomers saw stars with ears*, 语法在表 11.2 中。树中的非终结节点标上了子树的概率

表 11.2 一个简单的概率上下文无关文法(PCFG)的例子。非终结符是 S, NP, PP, VP, P 和 V。我们根据惯例把开始符 N' 写成 S。终结符是斜体词。这个表说明了语法规则及其概率。为了使得这种语法是 Chomsky 规范形式的, 我们采用了一些不太常见的 NP 规则, 它们会被用做本章后面的一个例子

$S \rightarrow NP\ VP$	1.0	$NP \rightarrow NP\ PP$	0.4
$PP \rightarrow P\ NP$	1.0	$NP \rightarrow \textit{astronomers}$	0.1
$VP \rightarrow V\ NP$	0.7	$NP \rightarrow \textit{ears}$	0.18
$VP \rightarrow VP\ PP$	0.3	$NP \rightarrow \textit{saw}$	0.04
$P \rightarrow \textit{with}$	1.0	$NP \rightarrow \textit{stars}$	0.18
$V \rightarrow \textit{saw}$	1.0	$NP \rightarrow \textit{telescopes}$	0.1

这个模型的假设是什么? 我们需要的条件是:

- 位置不变性。一棵子树的概率并不依赖于词串中它所支配的词语所处的位置(类似于 HMM 中的时不变性)。

$$\forall k \quad P(N_{k(k+c)}^j \rightarrow \zeta) \text{ is the same} \quad (11.6)$$

- 上下文无关性。子树的概率与不被子树支配的词无关。

$$P(N_{kl}^j \rightarrow \zeta | \text{anything outside } k \text{ through } l) = P(N_{kl}^j \rightarrow \zeta) \quad (11.7)$$

- 祖先无关性。子树的概率和子树之外的祖先节点无关。

$$P(N_{kl}^j \rightarrow \zeta | \text{any ancestor nodes outside } N_{kl}^j) = P(N_{kl}^j \rightarrow \zeta) \quad (11.8)$$

利用这些条件我们可以证明一棵树的概率可以通过与所用规则的概率相乘计算得到。但是我们需要能够区别不同非终结符的标记。因此,使用 N^j 中的左上标作为一个非终结符特定标记的识别索引。下面就是一个例子:

$$\begin{aligned}
 & P \left(\begin{array}{c} {}^1S \\ \swarrow \quad \searrow \\ {}^2NP \quad {}^3VP \\ \swarrow \quad \searrow \quad | \\ the \quad man \quad snores \end{array} \right) \\
 &= P({}^1S_{13} \rightarrow {}^2NP_{12} {}^3VP_{33}, {}^2NP_{12} \rightarrow the_1 man_2, {}^3VP_{33} \rightarrow snores_3) \\
 &= P({}^1S_{13} \rightarrow {}^2NP_{12} {}^3VP_{33}) P({}^2NP_{12} \rightarrow the_1 man_2 | {}^1S_{13} \rightarrow {}^2NP_{12} {}^3VP_{33}) \\
 &\quad P({}^3VP_{33} \rightarrow snores_3 | {}^1S_{13} \rightarrow {}^2NP_{12} {}^3VP_{33}, {}^2NP_{12} \rightarrow the_1 man_2) \\
 &= P({}^1S_{13} \rightarrow {}^2NP_{12} {}^3VP_{33}) P({}^2NP_{12} \rightarrow the_1 man_2) P({}^3VP_{33} \rightarrow snores_3) \\
 &= P(S \rightarrow NP VP) P(NP \rightarrow the man) P(VP \rightarrow snores)
 \end{aligned}$$

这里,通过链式规则扩展概率以后,我们可以首先加上上下文无关假设,然后加上位置不变性假设。

11.1 概率上下文无关文法的一些特征

这里我们给出一些使用 PCFG 的理由及其局限性:

- 为了覆盖一个庞大而且变化多端的文本语料库中的语言现象,需要不断扩展语法,这样就会使语法出现越来越多的歧义,因而对于大部分词序列开始出现很多结构不同的分析。一个 PCFG 给出了计算这些不同分析的合理度的一些思路。
- 一个 PCFG 并没有给出一个计算不同分析的合理度的好思路,因为它的概率估计是基于纯粹的结构因素的,并没有考虑词汇的同现因素。
- PCFG 有利于语法归纳。Gold(1967)表明 CFG 必须利用反例(不符合语法的例子)来学习(在有限识别的意义上,即,如果可以看见尽可能多的由语法产生的数据,我们是否可以识别一个语法)。但是 PCFG 可以只从正例中学习(Horning 1969)。从零开始做语法归纳仍然很困难,这是一个尚未解决的问题,因此把重点都放在了有括号标记的语料库(bracketed corpora)的学习上,我们将在第 12 章中讨论有关内容。
- 鲁棒性。真实的文本一般都会有语法错误,不流畅等问题。这个问题在某种程度上可以利用一个 PCFG 来避免,赋予这些不合适的句子一个很低的概率即可。

- PCFG 给出了一个英语的概率语言模型(然而 CFG 没有)。
- 如果用熵来衡量,那么一个 PCFG 的预测能力要比一个有相同参数的有限状态语法(例如,一个 HMM)更强。对于这种比较,我们计算参数的数目如下。一个有 V 个终结符, n 个非终结符的 PCFG 有 $n^3 + nV$ 个参数,但是一个有 K 个状态, M 个输出的 HMM 有 $K^2 + MK$ 个参数。虽然在这个 PCFG 中指数更高,但是使用的非终结符数目通常很少。参见 Lari and Young(1990)中对于特定人工语法的讨论。
- 在实践中,对于英语而言 PCFG 是一个比 n 元语法模型描述能力更差的语言模型(对于 $n > 1$)。一个 n 元语法模型可以考虑局部词汇上下文,但是一个 PCFG 却没有利用它。
- 就 PCFG 本身而言,它并不是很好的模型,但是我们希望把一个 PCFG 和三元语法模型结合起来。把一个 PCFG 规则建立在三元词组(以及树的一些附加的上下文敏感知识)之上的早期实验由 Magerman and Marcus(1991)以及 Magerman and Weir(1992)提出。我们在第 12 章中讨论了更好的解决方案。
- PCFG 有一些不合适的偏置。在其他条件相等的情况下,在一个 PCFG 中,一棵小树的概率比一棵大树大。这并不完全是没有道理的,它与 Frazier(1978)的最小附着启发是一致的,但是对于实际句子而言这并不是一个明智的模型,实际情况是中等长度的句子出现的频率最高。例如,表 4.3 表明华尔街日报中最常见的句子长度大约是 23 个词。PCFG 对于很短的句子给出了过大的概率。类似地,在其他条件相等的情况下,PCFG 分析中扩展很少的非终结符会比有很多扩展的非终结符更容易得到重视,因为较少重写会获得较高的概率(参见习题 12.3)。

PCFG 是否定义了一个语言模型是值得我们深入讨论的问题。最初我们认为,既然这些规则都符合式(11.3),那么就应该有 $\sum_{\omega \in \mathcal{L}} P(\omega) = \sum_t P(t) = 1$ 。但是,实际上这仅仅当规则概率是在有限推导中累积起来时才是正确的。例如,考虑下面的语法:

$$\begin{aligned} S &\rightarrow \text{rhubarb} & P &= \frac{1}{3} \\ S &\rightarrow SS & P &= \frac{2}{3} \end{aligned} \quad (11.9)$$

这个语法会生成所有的词串 $\text{rhubarb} \cdots \text{rhubarb}$ 。但是,我们发现这些词串的概率是:

$$\begin{aligned} \text{rhubarb} & \frac{1}{3} \\ \text{rhubarb rhubarb} & \frac{2}{3} \times \frac{1}{3} \times \frac{1}{3} = \frac{2}{27} \\ \text{rhubarb rhubarb rhubarb} & \left(\frac{2}{3}\right)^2 \times \left(\frac{1}{3}\right)^3 \times 2 = \frac{8}{243} \\ \dots & \end{aligned} \quad (11.10)$$

这个语言的概率是无限序列 $\frac{1}{3} + \frac{2}{27} + \frac{8}{243} + \cdots$ 的和,最终收敛到 $\frac{1}{2}$ 。因此不生成这个语言词串的一些树结构占据了一半的概率空间!这种分布在概率文献中被称为不一致性(inconsistent),但是这个词在与 NLP 相关的其他相关领域中已经有了非常不同的含义,因此我们称之为不拘(improper)分布。在实践中,不拘分布并不造成影响,特别是在我们主要只是比较不同概率估计量的时候。而且,从分析过的训练语料库中估计 PCFG 参数时(参见第 12 章),Chi and Geman(1998)表明我们总是能得到正确的概率分布。

11.2 概率上下文无关文法的问题

与 HMM 一样,PCFG 也有三个基本问题需要我们解决:

- 对应于一个语法 G , 句子 w_{1m} 的概率 $P(w_{1m} | G)$ 是什么?
- 句子最可能的分析是什么: $\arg \max_t P(t | w_{1m}, G)$?
- 我们怎样选择语法 G 的规则概率才能使一个句子的概率最大: $\arg \max_G P(w_{1m} | G)$?

在这一章中,我们只考虑满足 Chomsky 规范形式语法的例子,这种语法只含有下面形式的一元和二元规则:

$$\begin{aligned} N^i &\rightarrow N^j N^k \\ N^i &\rightarrow w^j \end{aligned}$$

在 Chomsky 规范形式中,一个 PCFG 的参数是:

$$\begin{aligned} P(N^j \rightarrow N^r N^s | G) & \text{ 如果有 } n \text{ 个非终结节点, 则有一个 } n^3 \text{ 的参数矩阵} \\ P(N^j \rightarrow w^k | G) & \text{ 如果有 } V \text{ 个终结节点, 则有 } nV \text{ 个参数} \end{aligned} \quad (11.11)$$

For $j = 1, \dots, n$,

$$\sum_{r,s} P(N^j \rightarrow N^r N^s) + \sum_k P(N^j \rightarrow w^k) = 1 \quad (11.12)$$

这些约束看上去符合表 11.2 中的语法(按照惯例所有不出现的概率都是 0)。任何一个 CFG 都可以用一个 Chomsky 规范形式的弱等同 CFG 来表示^①。

我们可以很有效地计算 PCFG 的概率,方法如下:先从 HMM 到概率正则文法,然后再从概率正则文法到 PCFG。考虑有如下形式规则的概率正则文法(PRG):

$$N^i \rightarrow w^j N^k \text{ or } N^i \rightarrow w^j \text{ and start state } N^1$$

这与 HMM 有点类似,但 HMM 是基于某个有限长度来定义词串的概率分布的,这可以看成是它们之间的区别:

$$\forall n \quad \sum_{w_{1n}} P(w_{1n}) = 1$$

但是,在一个 PCFG 或 PRG 中,在语言 $\mathcal{L}$ 中的所有词串集合上有一个概率分布:

$$\sum_{\omega \in \mathcal{L}} P(\omega) = 1$$

我们以下面句子的概率为例说明它们之间的差别:

$$P(\text{John decided to bake a})$$

在 HMM 中这个句子具有一个很高的概率,因为这很可能是一个句子的开头,但是在 PRG 或 PCFG 中它的概率会非常低,因为该句子是不完整的。

PRG 与 HMM 之间的大致关系可以用图 11.2 来表示。其中增加的初始状态以及从该状态

① 如果两个语法 G_1 和 G_2 都生成了同样的语言 $\mathcal{L}$ (具有同样的句子概率),那么它们是弱等同的。如果还为句子分配了同样的树结构(以同样的概率),那么它们是强等同的。

到 HMM 状态的转移用于映射 HMM 中的初始概率 π 。当需要表达词串的终结时,我们把 HMM 连接到一个终结状态,通常称之为下沉状态(sink state),一旦进入这种状态就不会离开。对于每个 HMM 状态,我们可以选择继续在基本的 HMM 中运行或进入下沉状态,在 PRG 中我们将后一种情况解释为词串的结束。



图 11.2 一个概率正则文法(PRG)

这样就给出了 PRG 怎样与 HMM 联系在一起的基本思路。我们可以把 PRG 作为 HMM 来实现,其中的状态相当于非终结符,终结符相当于输出符号,对应关系如下图所示:

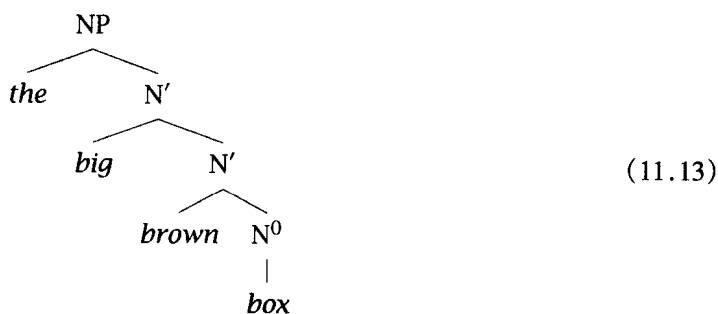
状态:	NP	→	N'	→	N'	→	N ⁰	→	下沉状态
输出:	<i>the</i>		<i>big</i>		<i>brown</i>		<i>box</i>		

回想一下在 HMM 中我们是怎样利用前向-后向概率来进行有效计算的,过程如下:

$$\text{前向概率 } \alpha_i(t) = P(w_{1(t-1)}, X_t = i)$$

$$\text{后向概率 } \beta_i(t) = P(w_{tT} | X_t = i)$$

现在我们考虑 PRG 分析,绘制成下面的树:



在树中,前向概率对应于一个特定的树节点以及之上任何节点的概率,后向概率对应于一个特定节点之下任何节点或词语的概率(给定节点)。于是便有了一种处理 PCFG 的更一般情况的方法。我们引入内部和外部概率,如图 11.3 中所示,定义如下:

$$\text{外部概率 } \alpha_j(p, q) = P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m} | G) \quad (11.14)$$

$$\text{内部概率 } \beta_j(p, q) = P(w_{pq} | N_{pq}^j, G) \quad (11.15)$$

内部概率 $\beta_j(p, q)$ 是以非终结符 N^j 开始,生成词串 $w_p \cdots w_q$ 的总概率。外部概率 $\alpha_j(p, q)$ 是以开始符 N^1 开始,生成非终结符以及 $w_p \cdots w_q$ 外部所有词语的总概率。

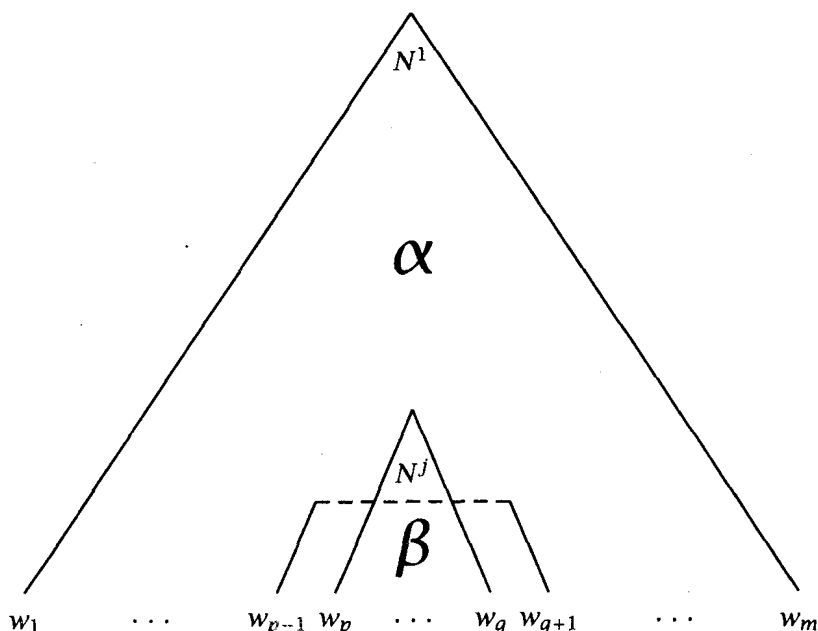


图 11.3 PCFG 中的内部和外部概率

11.3 词串概率的计算

11.3.1 内部概率的使用

通常,我们不能简单地通过对词串的所有可能分析树的概率求和来有效地计算词串的概率,因为计算量大多是指数级别的。计算一个词串的总概率的有效方法是内部算法(inside algorithm),它是一种基于内部概率的动态规划算法:

$$P(w_{1m}|G) = P(N^1 \xRightarrow{*} w_{1m}|G) \quad (11.16)$$

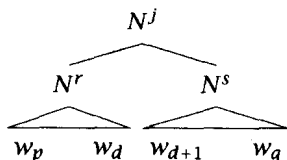
$$= P(w_{1m}|N^1_{1m}, G) = \beta_1(1, m) \quad (11.17)$$

子串的内部概率是根据词串子序列的长度归纳推导而计算得到的。

基本形式:我们希望得到 $\beta_j(k, k)$, 即规则 $N^j \rightarrow w_k$ 的概率大小:

$$\begin{aligned} \beta_j(k, k) &= P(w_k | N^j_{kk}, G) \\ &= P(N^j \rightarrow w_k | G) \end{aligned}$$

推导:我们希望得到 $p < q$ 时的 $\beta_j(p, q)$ 。因为我们是利用 Chomsky 规范文法进行推导的,所以第一条规则必须是 $N^j \rightarrow N^r N^s$, 因此可以利用归纳来求解,把词串在不同的位置分割成两部分,然后将结果相加:



于是,对于任意 $j, 1 \leq p < q \leq m$,

$$\begin{aligned}
 \beta_j(p, q) &= P(w_{pq} | N_{pq}^j, G) \\
 &= \sum_{r,s} \sum_{d=p}^{q-1} P(w_{pd}, N_{pd}^r, w_{(d+1)q}, N_{(d+1)q}^s | N_{pq}^j, G) \\
 &= \sum_{r,s} \sum_{d=p}^{q-1} P(N_{pd}^r, N_{(d+1)q}^s | N_{pq}^j, G) P(w_{pd} | N_{pd}^r, N_{(d+1)q}^s, G) \\
 &\quad \times P(w_{(d+1)q} | N_{pq}^j, N_{pd}^r, N_{(d+1)q}^s, w_{pd}, G) \\
 &= \sum_{r,s} \sum_{d=p}^{q-1} P(N_{pd}^r, N_{(d+1)q}^s | N_{pq}^j, G) P(w_{pd} | N_{pd}^r, G) \\
 &\quad \times P(w_{(d+1)q} | N_{(d+1)q}^s, G) \\
 &= \sum_{r,s} \sum_{d=p}^{q-1} P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)
 \end{aligned}$$

在上面的公式中,我们首先用链式规则把它们进行分割,然后利用 PCFG 的上下文无关假设,再用内部概率的定义重写了结果。利用这些重现关系,内部概率可以有效地自下而上计算。

例 2 上面的等式从表面上看很复杂,但是内部概率的计算实际上是简单直接的。可以通过改变两个小成分的标号以及覆盖的词语找到构造出一个特定成分的所有途径。表 11.3 给出了使用表 11.2 中的语法和图 11.1 中的句子的内部概率的计算。这些计算用一个分析三角形来表示。

表 11.3 内部概率计算。表中单元格 (p, q) 表明了通过内部算法计算出来的非零概率 $\beta'(p, q)$ 。内部概率的递归计算从对角线开始,然后移到右上角。
对于表 11.2 中的简单语法,惟一稍微复杂的情况是单元格 $(2, 5)$, 计算如下: $P(VP \rightarrow V NP) \beta_V(2, 2) \beta_{NP}(3, 5) + P(VP \rightarrow VP PP) \beta_{VP}(2, 3) \beta_{PP}(4, 5)$

	1	2	3	4	5
1	$\beta_{NP} = 0.1$		$\beta_S = 0.0126$		$\beta_S = 0.001\ 5876$
2		$\beta_{NP} = 0.04$ $\beta_V = 1.0$	$\beta_{VP} = 0.126$		$\beta_{VP} = 0.015\ 876$
3			$\beta_{NP} = 0.18$		$\beta_{NP} = 0.012\ 96$
4				$\beta_P = 1.0$	$\beta_{PP} = 0.18$
5					$\beta_{NP} = 0.18$
	astronomers	saw	stars	with	ears

▼ 使用这个例子中的语法和句子的更深入的计算留给读者作为习题。

11.3.2 外部概率的使用

我们也可以利用外部概率来计算词串的概率。对于任意 $k, 1 \leq k \leq m$,

$$\begin{aligned}
 P(w_{1m}|G) &= \sum_j P(w_{1(k-1)}, w_k, w_{(k+1)m}, N_{kk}^j | G) \\
 &= \sum_j P(w_{1(k-1)}, N_{kk}^j, w_{(k+1)m} | G)
 \end{aligned} \tag{11.18}$$

$$\begin{aligned}
 &\quad \times P(w_k | w_{1(k-1)}, N_{kk}^j, w_{(k+1)m}, G) \\
 &= \sum_j \alpha_j(k, k) P(N^j \rightarrow w_k)
 \end{aligned} \tag{11.19}$$

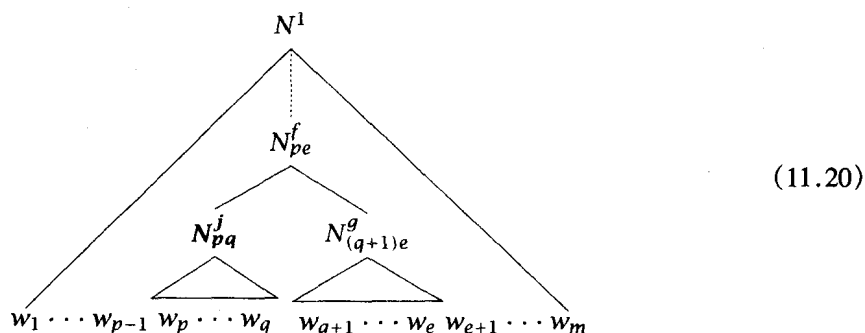
外部概率是自上而下计算的。我们可以看到,外部概率的推导计算需要参考内部概率,因此我们把使用外部算法(outside algorithm)来计算外部概率这一步放在后面。

基本形式:基本形式就是非终结符 N^i 作为根节点时的概率:

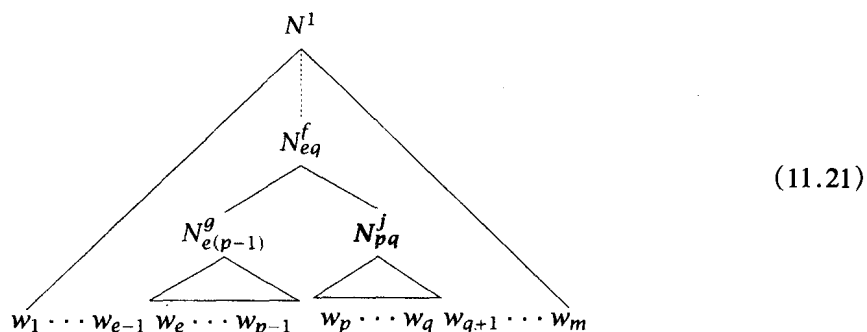
$$\alpha_1(1, m) = 1$$

$$\alpha_j(1, m) = 0 \quad \text{当 } j \neq 1$$

推导:依据前一步的派生,我们考虑节点可以在父节点左分支的情况:



或者可以在父节点右分支的情况:



对两个概率求和,但是必须把第一个和限制在 $g \neq j$ 的条件下,以避免重复计算形如 $X \rightarrow N^i N^j$ 的规则:

$$\begin{aligned}
\alpha_j(p, q) &= \left[\sum_f \sum_{e=q+1}^m P(w_{1(p-1)}, w_{(q+1)m}, N_{pe}^f, N_{pq}^j, N_{(q+1)e}^g) \right] \\
&\quad + \left[\sum_{f,g} \sum_{e=1}^{p-1} P(w_{1(p-1)}, w_{(q+1)m}, N_{eq}^f, N_{e(p-1)}^g, N_{pq}^j) \right] \\
&= \left[\sum_f \sum_{e=q+1}^m P(w_{1(p-1)}, w_{(e+1)m}, N_{pe}^f) P(N_{pq}^j, N_{(q+1)e}^g | N_{pe}^f) \right. \\
&\quad \times P(w_{(q+1)e} | N_{(q+1)e}^g) \left. \right] + \left[\sum_{f,g} \sum_{e=1}^{p-1} P(w_{1(e-1)}, w_{(q+1)m}, N_{eq}^f) \right. \\
&\quad \times P(N_{e(p-1)}^g, N_{pq}^j | N_{eq}^f) P(w_{e(p-1)} | N_{e(p-1)}^g) \left. \right] \\
&= \left[\sum_f \sum_{e=q+1}^m \alpha_f(p, e) P(N^f \rightarrow N^j N^g) \beta_g(q+1, e) \right] \\
&\quad + \left[\sum_{f,g} \sum_{e=1}^{p-1} \alpha_f(e, q) P(N^f \rightarrow N^g N^j) \beta_g(e, p-1) \right]
\end{aligned}$$

类似 HMM, 我们可以把内部概率和外部概率相乘:

$$\begin{aligned}
\alpha_j(p, q) \beta_j(p, q) &= P(w_{1(p-1)}, N_{pq}^j, w_{(q+1)m} | G) P(w_{pq} | N_{pq}^j, G) \\
&= P(w_{1m}, N_{pq}^j | G)
\end{aligned}$$

这个时候, 我们有必要假定一个非终结符 (然而在 HMM 中这无关紧要, 因为在每个时间点我们都处于某个特定状态)。因此, 句子以及覆盖词语 p 和 q 的成分的概率由下式给出:

$$P(w_{1m}, N_{pq} | G) = \sum_j \alpha_j(p, q) \beta_j(p, q) \quad (11.22)$$

但是, 我们知道总存在着非终结符, 覆盖整个树和每个终结符 (在 Chomsky 规范文法中)。其惟一子节点是单独终结符的非终结符节点, 称为预终结节点。因此, 求得整个词串概率的等式为式(11.17)和式(11.19)。式(11.17)等同于 $\alpha_1(1, m) \beta_1(1, m)$, 并利用了根节点, 式(11.19)等同于 $\sum_j \alpha_j(k, k) \beta_j(k, k)$, 并利用了每个词 w_k 之上必定有预终结符 N^j 的事实。

11.3.3 确定句子的最佳句法分析结果

可以通过使用内部算法寻找总体最优来构建一个 Viterbi 风格的算法, 这个算法能够找到句子最可能的分析, 同时记下给出这个总体最优的规则。因为 PCFG 中的独立性假设, 所以它的计算复杂度和 HMM 同样是 $O(m^3 n^3)$ 。

HMM 中 Viterbi 算法的成功之处在于定义了累积器 $\delta_j(t)$, 它可以记录在 t 时刻从状态 j 出发通过路径的最高概率。通过参照 PRG 可以给出 HMM 和 PCFG 之间的关系。现在我们希望找到以某个非终结符为根, 并覆盖某个特定子词串的部分分析树的最高概率。我们将会保留符号 δ 并采用累积器的方法:

$$\delta_i(p, q) = \text{一棵子树 } N_{pq}^i \text{ 分析的最高内部概率}$$

使用动态规划算法, 我们可以采用下面的方法计算句子最可能的句法分析。初始步骤是为每

个叶子节点的一元生成规则分配一个概率。推导的时候,我们知道应用的第一条规则肯定是二元规则,但是现在要寻找一个最可能的规则,而不是找所有这样的规则,并且在推导过程中需要用变量 ψ 记录最可能的规则。

1. 初始化过程

$$\delta_i(p, p) = P(N^i \rightarrow w_p)$$

2. 推导过程

$$\delta_i(p, q) = \max_{\substack{1 \leq j, k \leq n \\ p \leq r < q}} P(N^i \rightarrow N^j N^k) \delta_j(p, r) \delta_k(r+1, q)$$

保存回溯路径

$$\psi_i(p, q) = \arg \max_{(j, k, r)} P(N^i \rightarrow N^j N^k) \delta_j(p, r) \delta_k(r+1, q)$$

3. 程序的终止和路径的读出(通过回溯)。我们的语法有一个初始符 N^1 , 以初始符开始的最可能分析的概率是^①:

$$P(\hat{t}) = \delta_1(1, m) \quad (11.23)$$

我们希望重构 $\hat{t}$ 这个最高概率的树。我们把 $\hat{t}$ 当成一个节点集 $\{\hat{X}_x\}$ 来实现重构, 下面将说明怎样来构造这个集合。既然这个语法有一个开始符, 那么树的根节点一定是 N^1_m 。接下来我们将说明一般情况下如何构造一个非终结符节点的左右子节点。递归地应用这一过程, 就可以重构出整个树结构。如果在 Viterbi 分析中出现 $X_x = N^i_{pq}$, 而且 $\psi_i(p, q) = (j, k, r)$, 则有:

$$\begin{aligned} \text{left}(\hat{X}_x) &= N^j_{pr} \\ \text{right}(\hat{X}_x) &= N^k_{(r+1)q} \end{aligned}$$

注意上面写了 ‘arg max’ 的地方很可能并不存在惟一的极大值。我们假定在这些情况下分析器只是随机选择一个最大的句法分析。实际上, 如果保留所有的结果, 会把问题变得相当复杂。

11.3.4 概率上下文无关文法的训练

训练一个 PCFG 的基本思路是语法学习或语法归纳。但这只是在限定的意义下, 即我们假定语法由一定数目的终结符和非终结符以及预先给定的开始符构成, 同时假设规则集合也是提前给定的。通常我们假定所有的重写规则都存在, 或者假定一些语法中预定义的结构, 例如, 如果一些非终结符为预终结节点, 那么它只能被重写为终结节点。训练这个语法的过程很简单, 即只需要在这个体系结构内找到分配给不同语法规则的最优概率。

参照 HMM 的训练方法, 我们构造了一个 EM 训练算法, 即内部-外部算法, 它允许我们在未标注的句子训练 PCFG 的参数。基本假设是, 一个定义良好的语法可能会生成训练集中的所有句子, 因此需要找到最大化训练数据似然性的语法。我们首先基于单个句子进行训

① 我们也可以寻找支配整个句子的最高概率节点, 这个节点可以是任意类别的:

$$P(\hat{t}) = \max_{1 \leq i \leq n} \delta_i(1, m)$$

练,然后通过假定句子之间的独立性扩展到一个含有大量句子的大规模真实训练语料库上。

要确定规则概率,需要计算的是:

$$\hat{P}(N^j \rightarrow \zeta) = \frac{C(N^j \rightarrow \zeta)}{\sum_y C(N^j \rightarrow y)}$$

其中 $C(\cdot)$ 表示一个特定规则使用次数的计数。如果可以获得一个句法分析语料库,那么我们可以直接计算概率(将在第 12 章中讨论)。但是,一般情况下很难得到一个分析过的语料库,于是就有一个隐藏数据的问题。我们希望确定的是规则的概率函数,但是只能得到句子的概率,因为我们不知道规则概率,所以没有办法计算相对频率,因此采用迭代算法来改进估计。我们从一个特定的语法拓扑结构开始,这个结构指明了终结符和非终结符的个数以及一些规则的初始概率估计(也许只是随机选择的)。我们把训练句子的每个分析的概率作为它的置信度,然后对每条规则在每个位置使用的概率求和,以给出每条规则使用频率的期望。这些期望被用来更新我们对规则的概率估计,因此给定语法的训练集的似然性是增加的。

考虑:

$$\begin{aligned} \alpha_j(p, q) \beta_j(p, q) &= P(N^1 \xRightarrow{*} w_{1m}, N^j \xRightarrow{*} w_{pq} | G) \\ &= P(N^1 \xRightarrow{*} w_{1m} | G) P(N^j \xRightarrow{*} w_{pq} | N^1 \xRightarrow{*} w_{1m}, G) \end{aligned}$$

我们已经解决了怎样计算 $P(N^1 \xRightarrow{*} w_{1m})$ 的问题,并称这个概率为 π 。于是:

$$P(N^j \xRightarrow{*} w_{pq} | N^1 \xRightarrow{*} w_{1m}, G) = \frac{\alpha_j(p, q) \beta_j(p, q)}{\pi}$$

推导中使用非终结符 N^j 的次数的估计值是:

$$E(N^j \text{ is used in the derivation}) = \sum_{p=1}^m \sum_{q=p}^m \frac{\alpha_j(p, q) \beta_j(p, q)}{\pi} \quad (11.24)$$

在不是处理预终结符的情况下,我们用 β 的定义替换上面的概率,于是有 $\forall r, s, p < q$:

$$\begin{aligned} &P(N^j \rightarrow N^r N^s \mid N^1 \xRightarrow{*} w_{1m}, G) \\ &= \frac{\sum_{d=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{\pi} \end{aligned}$$

因此,对派生中特殊规则使用次数的估计可以通过对节点能够支配的所有范围的词语求和来得到:

$$\begin{aligned} &E(N^j \rightarrow N^r N^s, N^j \text{ used}) \\ &= \frac{\sum_{p=1}^{m-1} \sum_{q=p+1}^m \sum_{d=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{\pi} \quad (11.25) \end{aligned}$$

对于最大化步骤,我们有:

$$P(N^j \rightarrow N^r N^s) = \frac{E(N^j \rightarrow N^r N^s, N^j \text{ used})}{E(N^j \text{ used})}$$

因此,重估公式为:

$$\begin{aligned}\hat{P}(N^j \rightarrow N^r N^s) &= (11.25)/(11.24) \\ &= \frac{\sum_{p=1}^{m-1} \sum_{q=p+1}^m \sum_{d=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{\sum_{p=1}^m \sum_{q=p}^m \alpha_j(p, q) \beta_j(p, q)}\end{aligned}\quad (11.26)$$

同样,对于预终结符,

$$\begin{aligned}P(N^j \rightarrow w^k | N^1 \xRightarrow{*} w_{1m}, G) &= \frac{\sum_{h=1}^m \alpha_j(h, h) P(N^j \rightarrow w_h, w_h = w^k)}{\pi} \\ &= \frac{\sum_{h=1}^m \alpha_j(h, h) P(w_h = w^k) \beta_j(h, h)}{\pi}\end{aligned}$$

上面的 $P(w_h = w^k)$, 或者是 0, 或者是 1, 但是我们用第二种形式来表达, 力求和前面的情况尽可能相似。因此,

$$\hat{P}(N^j \rightarrow w^k) = \frac{\sum_{h=1}^m \alpha_j(h, h) P(w_h = w^k) \beta_j(h, h)}{\sum_{p=1}^m \sum_{q=p}^m \alpha_j(p, q) \beta_j(p, q)} \quad (11.27)$$

与 HMM 不同, 这时我们不可能避免处理多个训练实例的问题, 因为不能使用类似 HMM 中把多个实例连接起来形成一个大的实例的办法。假定我们有一个训练句子集合 $W = (W_1, \dots, W_\omega)$, 其中 $W_i = w_{i,1} \dots w_{i,m_i}$ 。让 f_i, g_i 和 h_i 分别成为用于分支节点、预终结符节点以及其他地方的非终结符, 现在从句子 W_i 计算:

$$\begin{aligned}f_i(p, q, j, r, s) &= \frac{\sum_{d=p}^{q-1} \alpha_j(p, q) P(N^j \rightarrow N^r N^s) \beta_r(p, d) \beta_s(d+1, q)}{P(N^1 \xRightarrow{*} W_i | G)} \\ g_i(h, j, k) &= \frac{\alpha_j(h, h) P(w_h = w^k) \beta_j(h, h)}{P(N^1 \xRightarrow{*} W_i | G)} \\ h_i(p, q, j) &= \frac{\alpha_j(p, q) \beta_j(p, q)}{P(N^1 \xRightarrow{*} W_i | G)}\end{aligned}\quad (11.28)$$

如果我们假定训练语料库中的句子是独立的, 那么对于语法而言训练语料库的似然性就等于其中句子概率的乘积。因此, 在重估过程中, 我们可以通过对多个句子的贡献求和来给出下面的重估公式。注意, 分母考虑了非终结符的所有扩展, 即它扩展为终结符或非终结符, 以满足式(11.3)中所有非终结符扩展概率之和为 1 的随机约束。

$$\hat{P}(N^j \rightarrow N^r N^s) = \frac{\sum_{i=1}^{\omega} \sum_{p=1}^{m_i-1} \sum_{q=p+1}^{m_i} f_i(p, q, j, r, s)}{\sum_{i=1}^{\omega} \sum_{p=1}^{m_i} \sum_{q=p}^{m_i} h_i(p, q, j)} \quad (11.29)$$

并且

$$\hat{P}(N^j \rightarrow w^k) = \frac{\sum_{i=1}^{\omega} \sum_{h=1}^{m_i} g_i(h, j, k)}{\sum_{i=1}^{\omega} \sum_{p=1}^{m_i} \sum_{q=p}^{m_i} h_i(p, q, j)} \quad (11.30)$$

内部-外部算法重复参数重估的过程, 一直持续到训练语料库的估计概率变化很小。如果 G_i 是训练中第 i 步迭代的语法(包括规则概率), 那么我们可以保证对应模型语料库的概率会增加, 而不会减小:

$$P(W | G_{i+1}) \geq P(W | G_i)$$

11.4 内部-外部算法的问题

但是,PCFG学习算法并非不存在问题:

1. 与类似 HMM 的线性模型比较,它的学习算法速度很慢。对于每个句子,训练中的每次迭代都是 $O(m^3 n^3)$ 的,其中 m 是句子的长度, n 是语法中非终结符的数目。
2. 局部极值是一个很严重的问题,Charniak(1993)指出,在每 300 个 PCFG 推导(随机初始化参数,使用一个简单的 PCFG 生成的数据)实验中就会得到一个不同的局部极值。换句话说,算法对于初始化参数非常敏感。这也许是实验其他学习方法的一个好地方。例如,模拟退火过程结合神经网络成功地避免了阻塞在局部极值的一些问题(Kirkpatrick et al. 1983; Ackley et al. 1985),但是对于大规模的 PCFG 而言,计算量还是太大。其他的部分解决方案是通过将一些参数初始化为 0 或进行语法最小化来添加限制,或者把非终结符从“贪婪的”终结符中分离出来重新分配。这些方法在 Lari and Young(1990)中进行了讨论。
3. 在人工语言实验的基础上,Lari and Young(1990)表明与理论上描述语言所需的非终结符数目相比,语法学习如果要取得满意的效果可能会需要更多的非终结符。在实验中,典型的情况是,从由 n 个非终结符的语法生成的训练文本中满意地学习一个语法,需要 $3n$ 个非终结符。这也包含了第一个问题。
4. 虽然算法保证提高训练语料库的概率,但是并不能保证算法学习所得的非终结符与这类通常由语言学分析得出的非终结符(NP 或 VP 等)有满意的相似。即使我们使用语言学家们熟悉的语法来初始化训练过程,训练机制也会完全按照它认为最好的非终结符类别做出修改。随着条件逐步确立,惟一严格的约束就是使 N^1 必须保持为初始符。一种选择是把更进一步的约束加在语法本质上。例如,我们可以把非终结符的功能专门化,以便它们只生成终结符或非终结符。使用这种形式的语法实际上简化了我们上面提出的重估方程。

因此,虽然对于 PCFG 在未经过标注的语料库中进行语法推导,从原则上讲是可行的,但是在实践中却极其困难。下一章从不同的角度讨论了使用 vanilla PCFG 的许多限制。

11.5 深入阅读

对于强(弱)等同性的 Chomsky 范式以及把任意 CFG 转化成各种规范形式的深入讨论,可以在 Hopcroft and Ullman(1979)中找到。在 NLP 中使用 CFG 分析的一般方法能够在大部分 AI 或 NLP 教材中找到,例如 Allen(1995)。

概率 CFG 方面的研究工作最先开始于 20 世纪 60 年代末 70 年代初,当时它是一个研究的热点。Booth and Thomson(1973)以及 Booth(1969)像本章中一样定义了一个 PCFG。在其他结果中,他们提出了一些不能由一个 PCFG 生成的上下文无关语言的词串概率分布,并且为 PCFG 派生出了充分必要条件,以定义一个合适的概率分布。这个时期的其他工作包括(Grenander 1967),(Suppes 1970),(Huang and Fu 1971)以及一些博士论文(Horning 1969; Ellis 1969; Hutchins 1970)。概率理论中的树结构通常被认为是分支过程,在(Harris 1963)和(Sankoff 1971)中有所讨论。

在 20 世纪 70 年代,随机形式语言的工作大部分都停止了,只有语音领域中还有一些相关的 PCFG 研究,也只是作为一个偶尔尝试的替代模型。Baker(1979)引入了内部-外部算法,并且证明了它的收敛性。本章介绍的内容都来自(Lari and Young 1990)。这篇论文包含了内部-外部算法的复杂性证明,(Lari and Young 1991)则在此基础上有了进一步的发展。

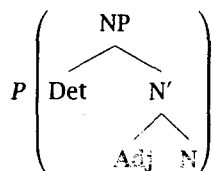
关于这里介绍的 PCFG 算法的扩展,可以参见(Charniak 1993)或(Kupiec 1991, 1992a)^①。Jelinek et al.(1990)和Jelinek et al.(1992a)提供了 PCFG 的完整介绍。特别需要指出的是,这些报告以及Jelinek and Lafferty(1991)和Stolcke(1995),提出了内部-Viterbi算法的从左至右增量的版本,这在一些语言模型的上下文中是非常有用的,例如语音识别。

在训练 PCFG 的章节中,我们假定了一个固定的语法结构。这自然会产生怎样确定这个结构以及如何自动学习的问题。在利用贝叶斯模型融合以及最小描述长度方法(Stolcke and Omohundro 1994b; Chen 1995),以自动确定合适的结构方面,已经做了一些工作,但是在现在的工作中通常仍然是利用语言学家的直觉来进行的。

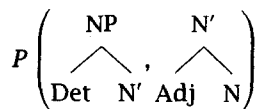
PCFG 也在生物信息学(例如,Sakakibara et al.1994)中用到,不过远没有HMM应用得广泛。

11.6 习题

11.1 [★★]考虑给定部分句子结构的(部分)语法分析树的概率:



一般来说,随着(子)树的增大,我们无法从任何现有的训练集中精确地估计这些树的概率(数据稀疏问题)。正如我们所见到的,PCFG从局部子树的联合概率中估计树的概率。



试问,这些局部子树的概率分布之间互相独立的假设是否合理?独立性假设允许我们把子树的概率估计成它所有局部树的概率的乘积。

使用一个有句法标注的语料库(例如Penn Treebank),并寻找一些共同的子树来说明独立性假设是否合理。如果不是,试问我们是否能找到一种联合局部子树的概率的方法,并且通过这种方法能得到较大子树概率的更好的经验估计。

11.2 [★]使用图 11.3 中的分析三角形,根据表 11.2 中的语法计算句子 *astronomers saw stars with ears* 的外部概率。从右上角开始沿着对角线工作。

① 对于熟悉chart分析的人而言,扩展是很直接的:在一个图表中,当我们移动点经过规则的时候构建最大的二元“遍历”。我们可以使用这种带有合适概率的虚拟语法来分析任意的 PCFG(完成构建一个成分的规则可以与初始规则有同样的概率,但是其他的规则概率都是 1)。

- 11.3 [★]使用图 11.3 和习题 11.2 中得出的句子 *astronomers saw stars with ears* 的内部和外部概率,通过迭代一次内部-外部算法重新估计表 11.2 中的语法概率。首先把图 11.3 中的内部概率与特定的规则以及用来得到内部概率的子树联系起来是很有帮助的。如果继续迭代内部-外部算法,规则概率将会收敛到什么值呢?为什么?
- 11.4 [★★★]记录类似于图 11.3 的分析三角中的可能节点覆盖是分析 CFG (Younger 1967; Hopcroft and Ullman 1979) 中 Cocke-Kasami-Younger (CKY) 算法的基本原理。作为一个练习,直接写出一个 CKY PCFG 分析器。我们也许需要把 Chomsky 规范文法形式的分析器扩展到更一般的上下文无关文法。一种方法是得到自己的通用形式,或者查询深入阅读中的其他相关论文。另一种方法是写一个语法转换,通过额外引入特别标明的必要节点,把一个 CFG 转换成一个 Chomsky 规范形式的 CFG,它可以在给定初始语法的情况下从输出中移走额外节点来表示分析树。如果把输入 CFG 严格限制在不包含任何空节点(扩展为空的非终结符)的情况下,这项任务是很容易的。
- 11.5 [★★★]较之于简单地分析一个词序列,如果把一个分析器与一个语音识别器连接起来,那么我们通常需要能够分析一个词语网格(word lattice),正如图 12.1 中展示的类型。扩展一个 PCFG 分析器,使得它可以工作在词语网格上。因为一个 PCFG 的运行时间决定于词语网格中词的数目,所以一个 PCFG 分析器在处理大型语音网格时是不实用的,但我们的 CPU 的速度每年都在加快!

第 12 章 概率句法分析

句法分析(parsing)是语块分析(chunking)思想的一个直接实现,语块分析通过识别出高层次的结构单元来简化句子的描述。从不同的句子中找到语块规律的一条途径是学习一种语法,这种语法能够解释我们所找到的分块结构。这属于语法归纳的范畴。目前学者们在语法归纳方面已经做了大量卓有成效的工作,例如怎样从没有标注的输入文本中学习结构,但是我们在这一章对它暂不做讨论。语法归纳技术对于有限状态语言是很好解释的,但是对于上下文无关或更加复杂的语言,其规模达到可以与人类语言复杂性相比的时候,归纳已经变得非常困难。从一个语料库中归纳出某些形式的结构也并不是很困难。一些语块分析算法,比如识别常见的子序列,会产生句子语块的一些表现形式,这些形式可以表示为一个短语结构树。但是,找到的这些表现形式与语言学或自然语言处理中所经常采用的短语结构之间通常并没有多大的相似之处。

迄今为止,在句法分析领域中存在很多争议,也许你会发现恰巧有人提出了与你正在努力研究的语法归纳程序偶然产生的相似的句法结构,而且这些也可能已经被当成了句法结构模型的证据。但是,这些找到的结构依赖于学习程序中隐含的归纳偏置。这也指明了另外一个方向,我们需要事先知道模型能够找到什么样的结构,同时应该首先确定我们对句子进行句法分析的目的。这里有各种可能的目的:使用句法结构作为语义解释的第一步;识别短语语块,为信息检索系统的索引服务;构建一个概率句法分析器作为一个优于 n 元语法的语言模型。这些问题的共同目标是构造这样一个系统:对于任意的句子都能够产生证明有用的结构,也就是要构建一个句法分析器。为了实现这个目标,我们没有必要一定从零开始,如果只是想要在产生有用的句法结构方面做一些有用的工作,那么可以使用所有的先验信息。我们在这一章中便采用了这种方法。

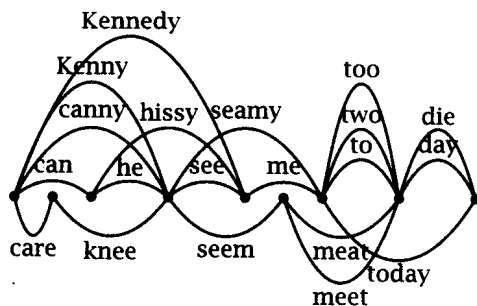


图 12.1 一个词语网络(已简化)

本章的内容分为两个部分。首先介绍了一些基本的概念、思路和一些相关的通用方法,这些方法在统计句法分析中很常见。第二部分展示了一些基于以上思路的实际句法分析系统,以及它们在实际中的性能。

12.1 一些概念

12.1.1 句法消歧分析

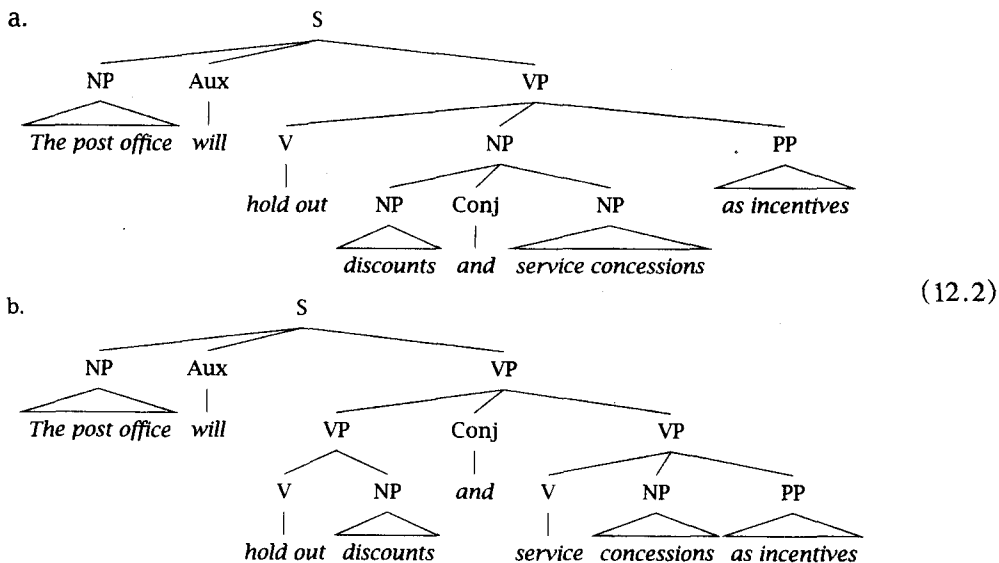
在句法分析器中至少有三种不同的途径可以使用概率：

1. **利用概率来确定句子** 一种可能的做法是将句法分析器看成是一个词语网格上的语言模型,用来确定什么样的词序列经过网格的时候会获得最大概率。在实际的应用比如语音识别中,实际的输入是不确定的,而且会有很多的假设,这些假设通常用词语网格(见图 12.1)来表示^①。在这里,句法分析器的任务是尽量去确定人说的话。最近采用这种方法的一个例子是(Chelba and Jelinek 1998)。
2. **利用概率来加速语法分析** 第二个目标是利用概率对句法分析器的搜索空间进行排序或剪枝。这使得句法分析器能够在不影响结果质量的情况下尽快找到最优的分析途径。最近研究这类有效方法的是(Caraballo and Charniak 1998)。
3. **利用概率选择句法分析结果** 句法分析器可以从输入句子的众多分析结果中选择可能性最大的。

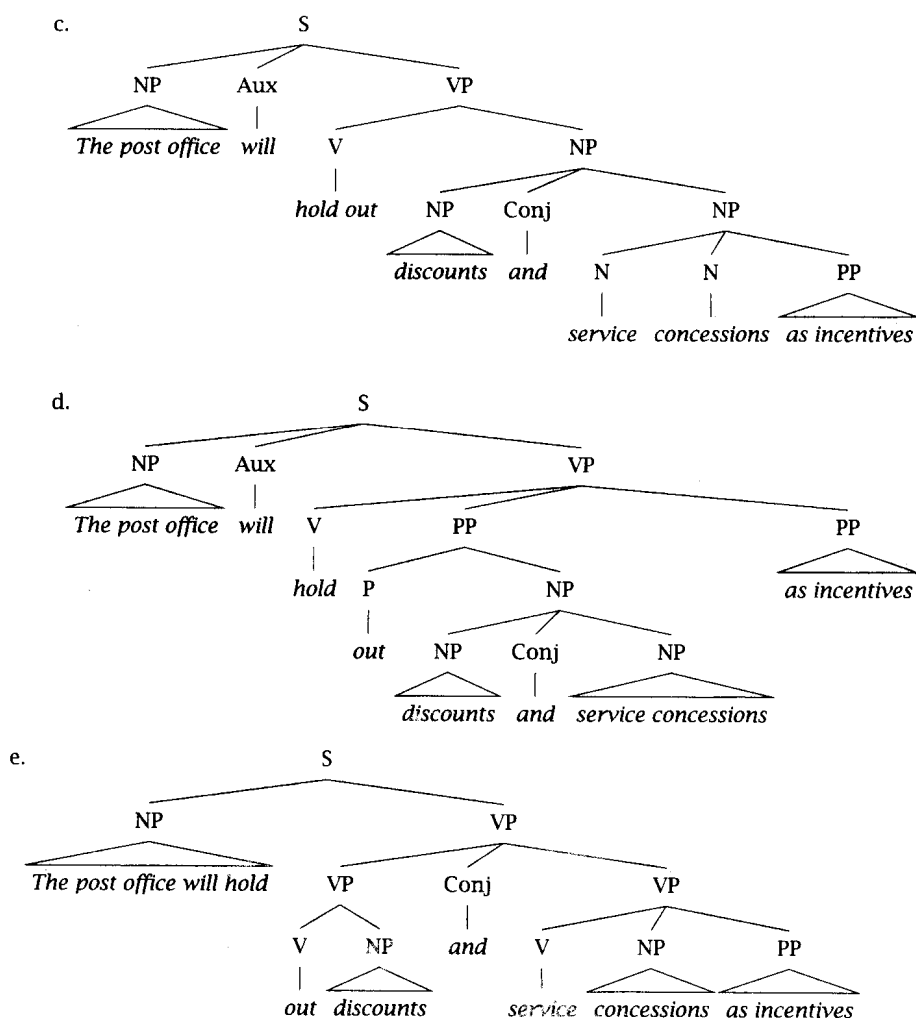
在本章中,我们重点讨论概率在句法分析树上的第三种应用:用统计句法分析器来进行消歧。

找到一个句子的树结构可以看成是实现消歧目标的关键,这也是我们在第1章中讨论的问题。例如,要判定例句(12.1)的确切含义,需要确定句子中的语义单元以及它们之间的关系。另外,歧义问题也是必须解决的问题,比如句子的正确分析是分析树(12.2a),分析树(12.2b),分析树(12.2c),分析树(12.2d)和分析树(12.2e)中的哪一个呢?

The post office will hold out discounts and service concessions as incentives. (12.1)



① 也可以用 n -best 表来表示,但是这样做会有不好的影响,它会使信号中不确定的离散区域的模糊性加倍。



人们在阅读计算语言学的书籍之后可能会很自然地产生这样一个想法:那些歧义现象都是很少见的或者是作者故意编造的,因为大部分书籍都含有同样的拗口的例子(比如关于钢笔和盒子,或者看见有人拿着望远镜)。但是,这仅仅是因为这些简短的例子很实用,所以才被普遍采用,实际上类似的歧义现象是很常见的。为了使例句(12.1)有一些新鲜感,我们采用了下面的方法:随机选择了一篇华尔街日报上的文章,用文章的第一个句子作为阐明我们观点的基础。要找到歧义并不困难^①。如果你仍不能信服歧义问题的严重性,可以在继续阅读下面的章节之前做一下习题 12.1。

该怎样处理这些歧义呢?在经典的直观方法中,一些歧义被看成是真正的语法歧义,句法分析器的任务是返回所有可能的结构。但是,有些奇怪的结果被看成是语法错误,为了减少荒谬的分析结果需要经常修改语法。例如,语法学家会觉得分析树(12.2d)应该被排除,因为

① 实际上我们对第一个句子做了一些精炼,因为像报纸中的很多句子一样,这个句子太长了。在一页中放下一个 38 个字的句子的分析树是很困难的。我们给出整个句子作为参考: *Postmaster General Anthony Frank, in a speech to a mailer's convention today, is expected to set a goal of having computer-readable bar codes on all business mail by 1995, holding out discounts and service concessions as incentives.*

hold 需要一个名词短语作为宾语, 动词 *hold* 加一个子类化框架可以保证这一点(详见后面的章节——译者注)。然而实际上这也不正确, 因为这个句法分析器将不能处理这样的句子: *The flood waters reached a height of 8 meters, but the sandbags held.*

比较而言, 一个具有统计学思想的语言学家对句法分析器能够为一个句子产生多少个分析结果不是非常感兴趣。通常句法分析中都会有一些限制分析结果数目的语法基础, 但是统计思想对于在允许的语法范围内的句法分析却没有什么限制。我们试图从那些符合句法结构的分析结果中分离出少数可能正确的结果, 但这几乎是不可能的。在许多情况下, 我们只关心“最好的分析结果”, 也就是被认为最有可能正确的结果。统计句法分析器一般会消除歧义, 并且给出一个比率来表示各种不同的分析作为实际分析的可能性。但是, 在传统的句法分析器中, 作为分析结果的句法分析树通常被送到下游的语义和世界知识模型中, 并由它们做出选择。一个统计句法分析器通常在运行的过程中使用各种扩展的词语符号和类别搭配来代替语义和世界知识, 以进行消歧。这至少也表明了词语的使用对我们了解语义有一定的作用。

12.1.2 树库

我们早先曾经提到过纯粹的语法归纳方法不会产生人们所需的句法分析树。对于这个问题, 一个显而易见的解决办法是给学习工具提供一些分析树的例子。这些分析树的一个集合称为树库(*freebank*)。这些经过正确句法分析的句子集合对于构建一个统计句法分析器是很有用的, 因此很多研究单位都研制了树库, 但是到目前为止, 从规模和可读性上来讲应用最为广泛的树库是 Penn 树库。

在图 12.2 中我们给出了 Penn 树库的一个例子。这个例子阐释了 Penn 树库中分析树的一些主要特性。分析树是用直接(Lisp)符号通过加括号表示的。组成短语的词都是很简单的(例如里面没有类似于短语 *Arizona real estate loans* 的复合名词消歧), 但是树库已经描述了当代语法中已知的主要短语类型。树库有时也尽量指明了语法或语义功能(图中的-SBJ 和-LOC 标签分别用来指明主语和位置词), 而且还利用空节点来指明已经理解的主语和抽取空隙。例如, 对于例子的状语从句中的已被理解的主语, 空节点已经被标上了*。在表 12.1 中, 我们总结了 Penn 树库中用到的短语类别(基本上遵从了第 3 章中讨论的类别)。

```
( (S (NP-SBJ The move)
  (VP followed
    (NP (NP a round)
      (PP of
        (NP (NP similar increases)
          (PP by
            (NP other lenders))
          (PP against
            (NP Arizona real estate loans))))))
    ,
    (S-ADV (NP-SBJ *)
      (VP reflecting
        (NP (NP a continuing decline)
          (PP-LOC in
            (NP that market))))))
  .))
```

图 12.2 Penn 树库中的一棵句法树

表 12.1 Penn 树库中短语类别的缩写。左边的一列是常见的类别。这些分类还包括了一些少见的古怪类别

S	简单从句(句子)	CONJP	多词联合短语
SBAR	带补语的 S'从句	FRAG	片段
SBARQ	<i>wh</i> -question S'从句	INTJ	感叹词
SQ	倒装 <i>Yes/No</i> 问题 S'从句	LST	表符号
SINV	倒装陈述 S'从句	NAC	非组成成分聚集
ADJP	形容词短语	NX	名词短语中的名词性成分
ADVP	副词短语	PRN	插入语
NP	名词短语	PRT	小品词
PP	介词短语	RRC	约简关系从句
QP	量词短语(名词短语内部)	UCP	不协调短语
VP	动词短语	X	不知道或不确定
WHNP	<i>wh</i> -名词短语	WHADJP	<i>wh</i> -形容词短语
WHPP	<i>wh</i> -介词短语	WHADVP	<i>wh</i> -副词短语

我们不得不提起的一个奇怪现象是,复杂名词短语用 NP-over-NP 结构表示。图 12.2 中的一个例子是以 *similar increases* 开头的名词短语。低层次的 NP 节点通常是指基本 NP(baseNP),它们只包含中心名词和前置内容,比如定语和形容词。高层次的 NP 节点(有时是两个)包含低层次的 NP 节点以及从属对象和修饰语。这种结构在当代的许多语法思想标准中是错误的,这些思想认为 NP 后置修饰词应归入某种 N'节点下的中心词,比定语的层次低(见 3.2.3 节)。另一方面,这种组织形式很好地抓住了 Abney(1991)所给出的语块(chunk)的定义,即中心名词及其前置修饰语看上去组成了一个语块,然而短语的后置修饰语是单独的语块。至少有些句法分析方面的工作直接采用了 Penn 树库的结构,把基本 NP(baseNP)当成句法分析中的一个单位。

即使是使用树库,在抽取示例分析中隐含的语法知识仍然会有归纳问题。但是,对于许多方法而言,这种归纳是无足轻重的。例如,要从一个树库中确定一个 PCFG,我们只需要计算局部分析树的频率,然后对它们进行归一化后给出概率。

许多人认为让语言学家构造树库比构造语法好得多,理由是对于独立的实际句子进行正确的分析比确定(直觉上感觉很庞大)特定的规则或句法结构的众多可能表现形式容易得多。从某种意义上来说这是正确的,因为一个语言学家即使绞尽脑汁也无法迅速想出所有的可能形式,但是为了能够构建树库必须至少假定一个隐含的语法。在多个人构建树库的项目中,把这个语法明晰化是很有必要的。Penn 树库的指导手册超过了 300 页。

12.1.3 句法分析模型与语言模型

句法分析的思想是能够根据某种语法 G 给出一个句子 s 的句法分析树。在概率句法分析中,我们以概率的形式表示每个分析结果的可能性,或者只返回最可能的分析结果。那么,最自然的想法就是定义一个概率句法分析模型,该模型对一个句子 s 给出分析树 t 的概率:

$$P(t|s, G) \quad \text{其中} \quad \sum_t P(t|s, G) = 1 \quad (12.3)$$

给定一个概率句法分析模型,分析器的任务就是找到句子 $\hat{t}$ 最可能的分析结果:

$$\hat{t} = \arg \max_t P(t|s, G) \quad (12.4)$$

这通常是很直接的,但有时因为实际操作中的原因,我们采用了各种启发式分析器或者采样分析器。这些方法在大多数情况下都能找到最可能的分析结果,但有时候却不能。

我们可以直接估计一个句法分析模型。研究者们已经构建过这类模型,但有点奇怪的是他们把概率建立在某个特殊的句子上。一般而言,我们需要把概率估计建立在一些更通用的数据类别上。最常用的方法是从定义一个语言模型开始,它为每个由语法生成的分析树都赋予了一个概率,那么我们就可以得到联合概率 $P(t, s|G)$ 。假设句子可由句法分析树确定(可由树的叶子节点恢复),如果 $\text{yield}(t) = s$,那么就变成 $P(t|G)$,否则为0。在这个模型下, $P(t|G)$ 是一个特定句子对应于语法 G 的特定分析结果的概率。下面我们把这个条件概率缩写为 $P(t)$ 。

在语言模型中,概率是针对整个语言 $\mathcal{L}$ 的,因此我们有:

$$\sum_{\{t: \text{yield}(t) \in \mathcal{L}\}} P(t) = 1 \quad (12.5)$$

整个句子的概率如下:

$$\begin{aligned} P(s) &= \sum_t P(s, t) \\ &= \sum_{\{t: \text{yield}(t)=s\}} P(t) \end{aligned} \quad (12.6)$$

这就意味着可以直接由一个语言模型构造一个句法分析模型。我们简单地用上述度量 $P(s)$ 去除语言模型中树的概率,那么最好的句法分析结果可以由下式给出:

$$\hat{t} = \arg \max_t P(t|s) = \arg \max_t \frac{P(t, s)}{P(s)} = \arg \max_t P(t, s) \quad (12.7)$$

因此,一个语言模型通常可以用做一个句法分析模型来选择各种句法分析结果。但是,语言模型也可以用做其他目的(例如,语音识别语言模型或估计一种语言的熵)。

另一方面,还没有有效的方法可以把任意的句法分析模型转化成语言模型。但是,注意我们在第11章讨论的 PCFG 句法分析模型中的一些偏置,IBM 的一些相关研究工作表明直接构造句法分析模型会比通过语言模型间接定义更好(Jelinek et al. 1994; Magerman 1995),而且直接定义的句法分析模型也已经被别人采用了(Collins 1996)。但是,在这项工作里,虽然总体的概率计算是建立在特定句子上的,但是分析结果的概率分解得到的原子概率并不依赖单独的句子,仍然是从整个训练语料库估计得到的。而且,Collins(1997)精炼了初始模型(Collins 1996),使得句法分析概率通过明晰的语言模型来定义,这样就在很大程度上增强了分析器的性能。因此,尽管语言模型对于句法分析模型来说并不是必要的,但是它们会提供一个更好的模型基础。

12.1.4 PCFG 独立性假设的弱化

上下文和独立性假设

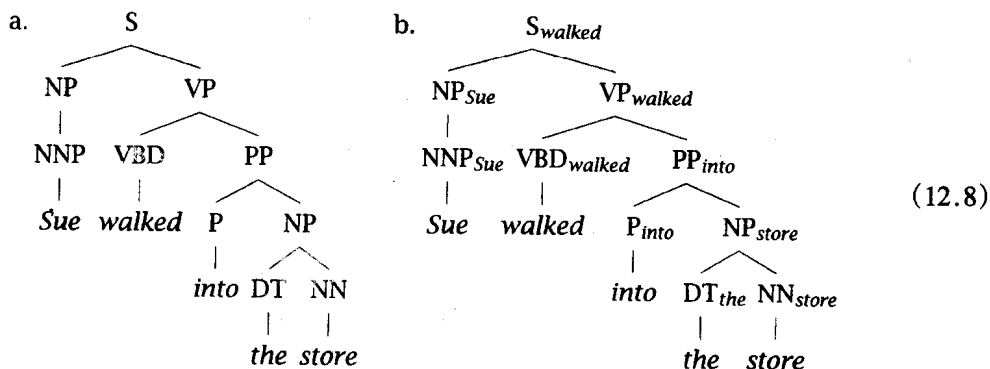
在语言理解的研究中,人们广泛接受这样一个观点,即在听人说话的时候利用言词中的上

下文消除语言歧义。上下文的利用形式有很多种,例如我们所处的位置(在电视机前还是在酒吧),我们在听谁说,或言词的时间先后关系。前面的言词会影响我们对后面句子的理解,这在心理学上称为启动效应(priming)。对句子语义直觉上的理解有时候会显得有点奇怪。而且,最近的很多研究工作都表明,人们在实际分析句子的时候利用了各种信息源^①。在以前定义的 PCFG 模型中,其独立性假设认为这些因素都与句法分析树的概率无关。但是,实际上以上所有信息源对于消除分析树的歧义都是相关的或有用的。即使我们没有把文本的上下文或者它的语义模型化,也能利用词语搭配近似这些信息源,以便帮助我们在局部进行语义消歧,或者把前文作为正文论述的一个指引(例如,我们可以从中检测文本的类型或主题)。因此,要构造一个优于 PCFG 的统计句法分析器,至少需要借助于这些信息源中的一部分。

词汇化

PCFG 中的独立性假设导致了两个方面的缺陷。最显著的一点是缺乏词汇化。在 PCFG 中,VP 扩展成为一个动词跟随两个名词短语的概率与参与的具体动词之间是独立的。这或许有些荒谬,因为这种可能性对于 *hand* 或 *tell* 之类的双宾语动词要比其他动词大得多。表 12.2 通过使用 Penn 树库中的数据,表明了各种普通的子类化框架的概率是怎样不同地依赖于 VP 中心动词的^②。这同时也说明了为什么我们在决定分析树结构的时候总希望包含实际句子中词语方面的更多信息。

即使在其他方面,词汇化的作用也很明显。一个显然的问题就是选择短语的附着位置。在第 8 章中我们详细讨论过,短语的词汇内容几乎总是能为确定正确的附着位置提供足够的信息,然而短语的语法结构类别一般只提供很少量的信息。标准的 PCFG 比 n 元语法模型差的一个方面就是它们完全没有利用词语之间的词汇依存关系。我们希望能够通过弥补这一点来得到一个比纯粹的线性词语级别 n 元语法模型更丰富的模型。最直接、最常用的词汇化 CFG 的方法是用中心词来标记每一个短语节点,因此分析树(12.8a)中的树将会词汇化成分析树(12.8b)中的树。



① 最后的这段话并不是毫无争议的。受乔姆斯基方法影响的心理语言学方面的工作一直认为人们首先构造句法分析,然后在消歧阶段对它们进行选择,例如(Frazier, 1978)。但是近期大量的工作,例如(Tanenhaus and Trueswell 1995, Pearlmuter and MacDonald 1992),表明这种观点并不正确,并且指出语义和上下文信息在句子理解阶段可以马上结合起来。

② 有人也许会怀疑在树库中出现一些概率很低但是非零的条目是错误的,但是要注意有这种现象存在:因为忽略了功能标记,一个类似于 *last week* 的时间 NP 可以出现在不及物动词的后面。

表 12.2 所选动词的常见子类化框架(VP 局部树扩展)频度。这些数据表明扩展 VP 所使用的规则强依赖于动词本身的特性。计数忽略了动词形式标记之间的差别。短语名称如表 12.1 所示,标记采用了 Penn 树库中的标记(见表 4.5 和表 4.6)

局部树	动词			
	<i>come</i>	<i>take</i>	<i>think</i>	<i>want</i>
VP → V	9.5%	2.6%	4.6%	5.7%
VP → V NP	1.1%	32.1%	0.2%	13.9%
VP → V PP	34.5%	3.1%	7.1%	0.3%
VP → V SBAR	6.6%	0.3%	73.0%	0.2%
VP → V S	2.2%	1.3%	4.8%	70.8%
VP → V NP S	0.1%	5.7%	0.0%	0.3%
VP → V PRT NP	0.3%	5.8%	0.0%	0.0%
VP → V PRT PP	6.1%	1.5%	0.2%	0.0%

词汇化模型的核心思想是认为中心词及其附属词之间存在着强依存,譬如中心名词和它的修饰形容词之间,或者动词和名词短语宾语之间,这里的名词短语宾语可以被中心名词近似。这通常是一个有效策略,但需要指出的是在非中心词对之间也存在依存。例如,例句(12.9)中的宾语 NP:

I got [NP the easier problem [of the two] [to solve]]. (12.9)

两个后置修饰语 *of the two* 和 *to solve* 都与前置修饰语 *easier* 有依存关系。它们的出现只很弱地依赖于 NP 中心词 *problem*。下面是这种类型的其他两个例子,中心词已经被加粗,涉及到非中心词依存的那些词是斜体的:

- a. Her approach was *more quickly* **understood** *than mine*.
 b. He lives in what must be the *farthest* **suburb** *from the university*. (12.10)

参见习题 8.16。

结构化上下文的概率依存

但是,单从结构上讲 PCFG 并非没有缺陷。PCFG 的内在思想是概率是上下文无关的。例如,一个名词短语以某种特定的方式扩展的概率与它在树中的位置无关。即使我们以某种方式词汇化 PCFG 来去掉其他的缺陷,结构上下文无关的假设依然存在。但是,这种语法假设实际上是非常错误的。例如,表 12.3 表明 Penn 树库中在主语位置还是在宾语位置展开一个 NP 节点,两个概率之间的差别很大。当 NP 含有后置修饰语时,代名词、专名词和限定 NP 在宾语位置很常见,单个名词在宾语位置则出现得很频繁。这就反映了主语通常表达了句子内部的主题这一事实。另举一个例子,表 12.4 比较了双宾语动词的第一个和第二个宾语 NP 的扩展。大家都知道第二个宾语一般不会是代名词,NP SBAR 趋向于扩展为第二个宾语则反映了句子中比较重要的部分出现在尾部的趋势,但是需要对语料库进行更全面的学习才能知道其他的一些影响。例如,我们现在并不了解为什么单个复数名词几乎不出现在第二个宾语的位置,但是它至少说明上下文的分布不是独立的。

上面的观察结果表明,我们能够通过更好地考虑词汇知识和上下文结构来构造一个比基于 PCFG 更好的概率句法分析器。伴随而来的挑战是(经常会有),在找到能够给予我们大量

额外鉴别信息的因素的同时,怎样才能不陷入数据稀疏问题的困境。本章第二部分中的系统将会依照这个思路给出一系列的方法。

**表 12.3 NP 通常扩展作为主语与作为宾语的对比,以对数比来排序。这些数
据表明用来扩展 NP 的规则强依赖于其父节点(作为主语或者宾语)**

扩展	作为主语的百分比	作为宾语的百分比
NP → PRP	13.7%	2.1%
NP → NNP	3.5%	0.9%
NP → DT NN	5.6%	4.6%
NP → NN	1.4%	2.8%
NP → NP SBAR	0.5%	2.6%
NP → NP PP	5.6%	14.1%

**表 12.4 NP 通常扩展作为 VP 内部的第一个宾语和第二个宾语的对比。这些数
据可以作为说明非终结符扩展中结构上下文信息重要性的另一个例子**

扩展	作为第一个宾语的百分比	作为第二个宾语的百分比
NP → NNS	7.5%	0.2%
NP → PRP	13.4%	0.9%
NP → NP PP	12.2%	14.4%
NP → DT NN	10.4%	13.3%
NP → NNP	4.5%	5.9%
NP → NN	3.9%	9.2%
NP → JJ NN	1.1%	10.4%
NP → NP SBAR	0.3%	5.1%

12.1.5 分析树概率和派生概率

在 PCFG 的框架下,通过把句法分析树的每棵局部子树的概率相乘就可以得到句法分析树的概率,局部子树的概率是由产生式规则给出的。这棵分析树可以被看成是在每个节点做出选择的时候,对树分支的简单记录,它仅仅只依靠节点的标号。正像我们在第 3 章中看到的那样,在生成语法模型中^①,由语法生成句子,通常是从一个起始符号开始执行一系列自上而下的重写规则,直到其所有叶节点都是终结符(也就是单词)为止。例如,图 12.3(a)说明了一个句子利用表 11.2 中的语法的派生过程,每一步非终结符都会根据语法而重写。重写系统概率化的一个直接方法是在每个派生节点上都定义概率分布。例如,在最后一步,我们把最终的 N 重写为 *telescopes*,但是根据语法还有其他选择。派生过程的线性步骤可以直接映射到一个标准的随机过程,其中的状态都是语法的产物。既然生成语法能够生成语言中的所有句子,那么一个派生模型本身也是一个语言模型。

因此,得到句法分析树概率的一条途径是依据派生概率。一般情况下分析树都有好几个派生形式。例如,分析图(12.11)中的分析树就不仅只有图 12.3(a)所示的形式,还有其他的形式,比如图 12.3(b),其中第二个 NP 在 V 之前被重写了。

^① 根据 Chomsky(1957)给出的原始含义;在最近的工作中 Chomsky 建议“生成”仅仅只表示“形式”(formal)(Chomsky 1995:162)。

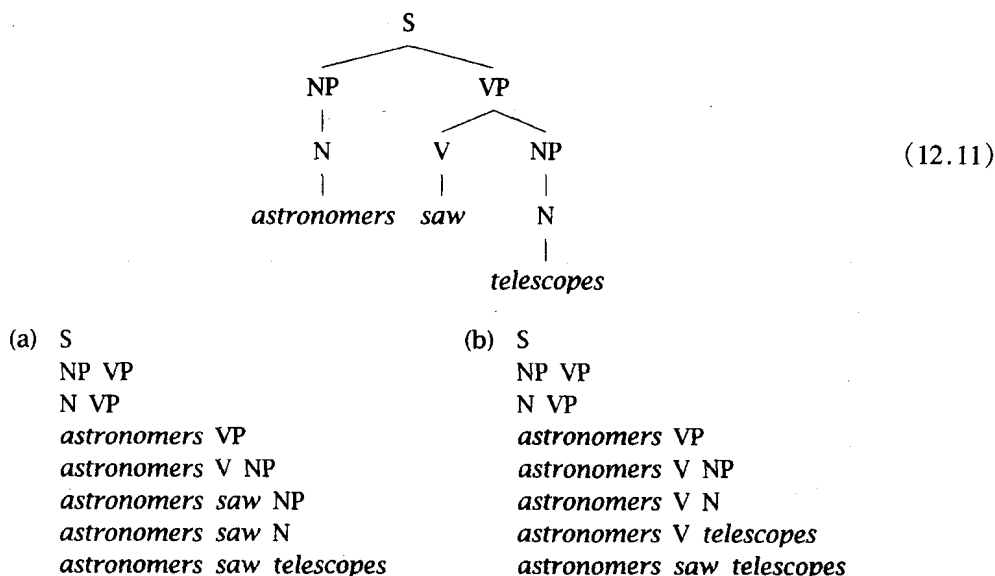


图 12.3 一棵树两个 CFG 派生

因此,为了估计一棵分析树的概率,一般情况下需要计算:

$$P(t) = \sum_{\{d: d \text{ is a derivation of } t\}} P(d) \quad (12.12)$$

不过,在许多例子中,譬如 PCFG,没有必要这么复杂。非常明显(虽然很难证明),在 PCFG 的例子中派生次序的选择对于最终的概率是没有影响的^①。无论在一次派生中选择哪个节点进行重写,无论我们假设的概率分布如何,分析树的最终概率都是一样的。因此我们可以设法为每棵树找到一个惟一的派生来实现简化,我们把这个派生称为规范派生(canonical derivation)。例如,对于图 12.3(a)中的最左派生,我们每一步都扩展最左的非终结符作为一个规范派生。如果这是可能的,我们就可以说:

$$P(t) = P(d) \quad \text{这里 } d \text{ 是 } t \text{ 的规范派生} \quad (12.13)$$

这种简化是否可行依赖于模型条件下的概率学性质。在 PCFG 中这是可能的,因为概率仅仅依赖于父节点,所以其他节点是否被改写都没有影响。如果利用了更多的上下文,或者有一些其他的方法来生成同样的结构块,那么分析树的概率就会在很大程度上依赖于派生次序。可以参见 12.2.1 节和 12.2.2 节^②。

我们把 α_u 重写成 α_v 的单独一步 r_i 表示成: $\alpha_u \xrightarrow{r_i} \alpha_v$ 。要计算一个派生的概率,可以使用链式规则,以前一步为条件给派生中的每一步都分配一个概率。对于一个标准的重写语法而言,有类似下面的形式:

$$P(d) = P(S \xrightarrow{r_1} \alpha_1 \xrightarrow{r_2} \alpha_2 \xrightarrow{r_3} \dots \xrightarrow{r_m} \alpha_m = s) = \prod_{i=1}^m P(r_i | r_1, \dots, r_{i-1}) \quad (12.14)$$

① 证明需要使用 Hopcroft and Ullman(1979)提出的树映射推导。

② 即使是在这些例子中,我们也会选择根据规范派生的概率估计来近似树的概率,但是这对性能可能会有不利影响。

我们可以考虑上面表达式中的条件,也就是已经被应用的重写规则,将其作为分析历史记为 h_i ,因此 $h_i = (r_1, \dots, r_{i-1})$ 。这就是 IBM 最先尝试的基于历史的语法 HBG。因为我们不能把整个历史模型化,所以通常通过等价函数 π 来形成等价历史类别,那么以上估计可以写成:

$$P(d) = \prod_{i=1}^m P(r_i | \pi(h_i)) \quad (12.15)$$

PCFG 是这个框架下的一个特例。PCFG 的等价函数只是简单地返回了短语构造器中其余的最左非终结符。因此, $\pi(h_i) = \pi(h'_i)$ 当且仅当 $\text{leftmost}_{\text{NT}}(\alpha_i) = \text{leftmost}_{\text{NT}}(\alpha'_i)$ 。

12.1.6 其他途径

在第 11 章中我们很自然地使用概率来扩展上下文无关文法,以至于也许有人会认为这是惟一的,或者说至少惟一明智的做法。术语 PCFG(概率上下文无关文法)的使用印证了这一点。但是,实际上概率文法并不是惟一的。与经典的上下文无关文法不一样的众多概率和分析方法都或强或弱地收敛到等价结果上,而不同的处理方法一般会导致不同的概率文法。从概率角度来讲,最重要的是不同事物的概率是建立在什么条件上的(从另一个角度看,即做出了什么样的独立性假设)。但是,概率文法有时候也是等同的。例如,HMM 从左向右或从右向左进行处理的结果都相同,如果基本条件改变了,那么就会有一个不同的概率文法,即使它含有同样的范畴基础。作为一个例子,我们在这里考虑另一个基于 CFG 的概率文法,概率左角文法(Probabilistic Left-Corner Grammars, PLCG)。

概率左角文法

如果我们考虑分析方式,那么 PCFG 属于自上而下分析的概率分析,因为在每一步预测子节点的时候都只给出了父节点的相关知识。而其他的分析思想提出了不同概率条件的模型,通常,这些条件是自上而下和自下而上的信息的混合。一种方法是左角分析策略。

左角句法分析器(Rosenkrantz and Lewis 1970; Demers 1977)结合了自下而上和自上而下的过程。它从一个目标类别(当前构造的根节点)开始,然后查看左角的字符串(例如,移进下一个终结符)。如果左角类别与目标类别是同一个类别,那么就可以停止了。否则,它就需要在语法中找到一条规则,并且应用这条规则从左角发射一个可能的局部分析树,该规则中右边的第一个元素便是左角类别。发射出的这个局部分析树中的其余子节点将会成为目标类别,而且每一个都将进行递归的左角分析。当局部树完成后,我们开始递归地以子树作为左角开始进行左角分析,目标类别与开始的目标类别相同。为了更精确地描述,图 12.4 中给出了产生一个简单的左角识别器的伪码^①。这个特殊的句法分析器假定这些词汇信息都是在规则的右边引入的,例如,类似于 $N \rightarrow \text{house}$,并且假定当水平写的时候栈顶对应规则的左边。这个分析器以查找和搜索的方式工作,搜索类别在栈中用类别上面加一横线表示。我们用 α 表示一个单独的终结符或非终结符(或者空字符串,如果我们希望在语法中保留空类别),用 γ 表示一个终结符或非终结符组成的序列(可能是空序列)。这个句法分析器有三种操作:移进(shifting)、发射(projecting)和附着(attachng)。同时我们将会为这些操作给出概率分布。什么时候移进是确定

^① 这里的表示借用了 Mark Johnson and Ed Stabler (1993)未发表的草稿。

的。如果栈顶是搜索类别 $\bar{C}$, 就必须移进, 在其他时候都不能成功移进。在移进的事物上也会有一个概率分布。在其他时刻, 我们必须确定是否附着或发射。比较有意思的一种情况是, 在左角类别和目标类别相同时确定是否附着。如果不附着, 就必须发射。最后我们需要在给定左角类别(lc)和目标类别(gc)的条件下得到发射一个特定的局部分析树的概率。在这个模型下, 我们对发射操作会有如下概率:

$$P(\text{SBAR} \rightarrow \text{IN } S | lc = \text{IN}, gc = S) = 0.25$$

$$P(\text{PP} \rightarrow \text{IN } NP | lc = \text{IN}, gc = S) = 0.55$$

```

1  comment: Initialization
2  Place the predicted start symbol  $\bar{S}$  on top of the stack
3  comment: Parser
4  while (an action is possible) do one of the following
5      actions
6      [Shift] Put the next input symbol on top of the stack
7      [Attach] If  $\alpha\bar{\alpha}$  is on top of the stack, remove both
8      [Project] If  $\alpha$  is on top of the stack and  $A \rightarrow \alpha \gamma$ , replace  $\alpha$  by  $\bar{\gamma}A$ 
9      endactions
10 end
11 comment: Termination
12 if empty(input)  $\wedge$  empty(stack)
13 then
14     exit success
15 else
16     exit failure
17 fi

```

图 12.4 一个左角栈句法分析器

为了产生一个语言模型来反映左角句法分析器的操作, 我们可以把分析的每一步操作看成是派生的每一步。换句话说, 我们可以用左角概率来生成一棵分析树。然后, 正如最后一部分所示, 我们可以按照分析树的左角派生概率来表示分析树的概率。在左角派生中, 每一棵分析树都对应一个惟一的派生, 因此有:

$$P_{lc}(t) = P_{lc}(d) \quad \text{其中 } d \text{ 是 } t \text{ 的 LC 派生}$$

一个句子的左角概率可以用通常的方法计算:

$$P_{lc}(s) = \sum_{\{t: \text{yield}(t)=s\}} P_{lc}(t)$$

派生概率可以表示成派生过程中每一个单独操作概率的乘积。假设 $(C_1, \dots, C_m)$ 是树 t 的 LC 句法分析派生过程 d 的操作序列。因此, 根据链式规则有:

$$P(t) = P(d) = \prod_{i=1}^m P(C_i | C_1, \dots, C_{i-1})$$

在实际操作中, 我们不可能把每一步分析决策的概率都建立在整个历史上。我们这里要设计最简单的左角模型, 因此假设每一步分析决策的概率在很大程度上并不依靠历史, 仅仅依

靠分析器的状态。特别地,我们假定它只依靠左角和分析栈栈顶的目标类别。

左角分析器的每个基本操作可以是移进、附着或左角发射。在上面所提到的独立性假设下,移进的概率在给定当前目标类别(gc)的条件下就是某个特定的左角子节点(lc)移进的概率,我们将其表示为 P_{shift} 。何时移进是确定的。如果一个目标类别(加上了横线)在栈顶(因此没有左角类别),那么就必须移进。如果不移进,就必须在附着和发射之间选择,附着的概率表示为 P_{att} 。如果左角和目标类别是一样的,那么附着只有非零概率,但我们对所有的对子(pair)都有定义。如果不附着,就以概率 P_{proj} 基于左角发射一个成分。因此,每一步基本操作 C_i 的概率都可以按照概率分布 P_{shift} , P_{att} 和 P_{proj} 来表达,如下所示:

$$P(C_i = \text{shift } lc) = \begin{cases} P_{shift}(lc|gc) & \text{如果栈顶是目标类别} \\ 0 & \text{其他} \end{cases} \quad (12.16)$$

$$P(C_i = \text{attach}) = \begin{cases} P_{att}(lc, gc) & \text{如果栈顶不是目标类别} \\ 0 & \text{其他} \end{cases} \quad (12.17)$$

$$P(C_i = \text{proj } A \rightarrow \gamma) = \begin{cases} (1 - P_{att}(lc, gc))P_{proj}(A \rightarrow \gamma|lc, gc) & \text{如果栈顶不是目标类别} \\ 0 & \text{其他} \end{cases} \quad (12.18)$$

这些操作遵循下面的限制条件:

$$\sum_{lc} P_{shift}(lc|gc) = 1 \quad (12.19)$$

$$\text{If } lc \neq gc, P_{att}(lc, gc) = 0 \quad (12.20)$$

$$\sum_{\{A \rightarrow \gamma: \gamma=lc \dots\}} P_{proj}(A \rightarrow \gamma|lc, gc) = 1 \quad (12.21)$$

从上面可以注意到,不同的移进和发射选择的概率之和为 1。因此,既然其他的概率互为补充,每个基本操作可以获得的动作的概率之和为 1。在派生中也没有死节点,因为除非 A 是 gc 的一个可能的左角成分, $P_{proj}(A \rightarrow \gamma | lc, gc) = 0$ 。因此这些概率定义了一个语言模型^①,即 $\sum_i P_{lc}(s | G) = 1$ 。

Manning and Carpenter(1997)对这类PLCG进行了一些初步尝试。尽管上面所用的独立性假设还是很极端的,但是可以得到比 PCFG 稍微好一些的概率模型,因为基本的左角分析动作都是建立在目标类别上的,而不仅仅是局部分析树的概率。例如, NP 的某个特定扩展概率在主语位置和宾语位置是不同的,因为它们的目标类别不一样。这样就可以体现出表 12.3 所示的分布差异^②。正因为如此, Manning and Carpenter(1997)说明了为什么 PLCG 比基本的 PCFG 优越。

其他方法

左角句法分析是一个有趣的例子。左角句法分析器从左向右工作,联合了自上而下和自下而上两种预测,在习题 12.6 中讨论的一般化左角句法分析器模型族中占据了很重要的地

① 根据有限树的概率积累,这个问题在第 11 章中讨论过。

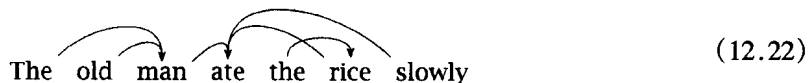
② 但是应当注意到这些在表 12.4 中是无法获得的。

位。不过它并不是在 CFG 分析算法上构造概率分析器的惟一可能方法,实际上还有其他方法。

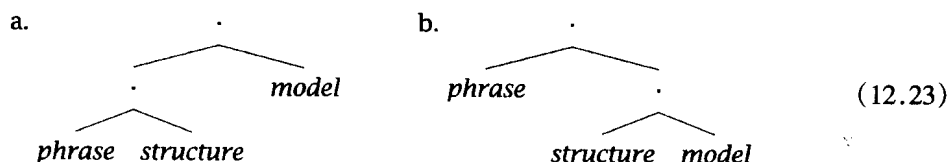
使用自下而上预测,移进-约减分析器很显然是个不错的选择。特别地,已经有人开始着眼于构造 Tomita(1991)通用 LR 分析方法的概率版本。Briscoe and Carroll(1993)做了这方面的初始研究,但是其模型在 LR 分析表中可能不合适地控制着一个合一句法分析器(unification-based parser),合一失败导致了概率分布没有被捕获的分析失败。Inui et al. (1997)中详细描述了一个稳健的概率 LR 句法分析器。

12.1.7 短语结构语法和依存语法

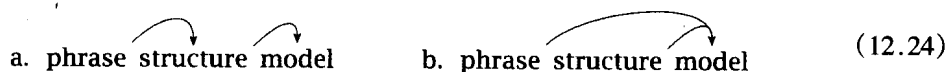
在现代语言学和自然语言处理中通常采用短语结构树来描述句子结构。但是,比较早的方法是采用词之间的依存关系来描述语言结构。这样的框架称为依存语法。在依存语法中,某个词是句子的中心词,其他的词或者依赖于这个词,或者依赖于那些通过一系列依存关系与中心词有联系的其他词。依存关系通常由带箭头的曲线表示,如下面的例子所示:



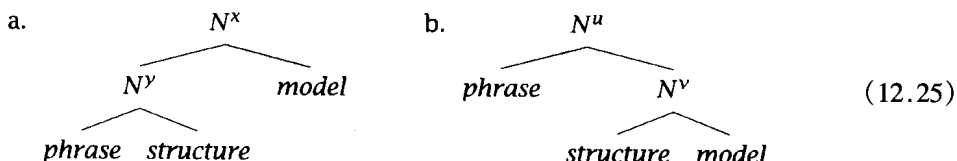
在统计 NLP 中按照依存关系来思考是很有用的,但是我们也希望了解短语结构和依存模型之间的关系。Lauer(1995a;1995b)研究了复合名词的结构消歧,认为依存模型比邻接模型好得多。假设我们需要对复合名词譬如短语结构模型进行消歧。以前的工作已经考虑到复合名词有两种可能的树结构,如分析树(12.23)所示,并且试图在两者之间做出选择,其判断的依据是在语料库中 $phrase \leftrightarrow structure$ 和 $structure \leftrightarrow model$ 哪个具有更紧密的搭配关系。



Lauer 认为应该根据依存结构检查歧义,如分析树(12.24)所示,很明显它们之间的差别是 $phrase$ 依赖于 $structure$ 还是依赖于 $model$ 。通过对比表明,依存模型比邻接模型更好一些。



Lauer 认为以前的工作是有瑕疵的,而且坚持认为依存模型的优势很明显。但是这个结果并没有表明依存模型相对于短语结构模型有根本的优势。邻接模型的问题如图 12.25 所示,在树的多次分解中,模型仅仅只考虑了节点 N^y 和 N^o ,而忽视了节点 N^x 和 N^u 。



如果要纠正邻接模型,那么必须考虑节点 N^x 和 N^u ,并对短语结构树做明显的词汇化,因

此 N' 将会用 *structure* 标注, N'' 将会用 *model* 标注(因为英语复合名词的中心词在右边), 我们很容易发现这两个模型是等同的。在词汇化 PCFG 类模型下, 我们会发现 $P(N') = P(N'')$, 因此确定概率的方法是比较 $P(N')$ 和 $P(N'')$ 。但是, 这等同于比较两种组合 $phrase \rightarrow structure$ 和 $phrase \rightarrow model$ 中哪一种组合搭配得更紧密。

各种类型的依存语法和对应类型的短语结构语法实际上是同构的。使用间接弧的依存语法与一个每条规则至少引入一个终结节点的短语结构语法是等同的。而更常见的使用直接弧的依存语法与 1-bar 层 X' 语法等同。即, 对于语法中的每个终结符 t , 都有一个非终结符 $\bar{t}$, 并且语法中惟一的规则形式是 $\bar{t} \rightarrow \alpha t \beta$, 这里的 α 和 β 是非终结符序列(可以为空)(参见 3.2.3 节)。依存语法的另一个通常做法是标明依赖性。这相当于不仅把每个局部子树的一个子节点标为中心词(在 X-bar 方案中已经隐含实现了), 而且标明了和每个子节点的关系。如果概率条件是一样的, 那么这些结果支持两种语法的概率版本^①。

不过, 依存语法已经被广泛应用在概率分析上, 它具备两个关键优势。其一, 我们以前讨论过词汇信息是解决大多数分析歧义的关键, 因为依存语法是直接按照词语之间的依存关系工作的, 直接依据词之间的依存关系又是歧义决策的依据。没有必要对句子建立一个庞大的超结构(superstructure)(即短语结构树), 也没有必要脱离句子中的词在结构上进行歧义决策。另外, 没有必要为如何词汇化一棵短语结构树的问题担忧, 因为没有把结构与句子中的词脱离开来。实际上, 依存语法的拥护者认为很多短语结构树的超结构都是冗余的: 对于一个句子的理解而言是没有必要的。

依存语法的第二个优势在于依存关系给出了分解短语结构规则的方法, 而且能够估计它们的概率。利用 Penn 树库归纳句法分析器时存在一个问题, 因为树都是非常平坦的, 所以尽管这种含有大量子节点的树的种类很少见, 但是它的数量却很多。在未知数据中, 一些树结构对我们来说可能是以前没有见过的, 所以如果想利用 PCFG 一次估计出所有局部树的概率, 有时不太可能。但是我们注意到, 使用依存语法可以通过单独估计每个主从关系的概率来分解一个局部树的概率。如果我们以前从未见过分析树(12.5)中的局部分析树, 那么在一个 PCFG 模型中最好是退回到一些默认的未见过的树的概率上。但是, 如果把树分解成依存关系, 就像图 12.5(b)中一样, 然后给出已知的类似图 12.5(c)或图 12.5(d)的树, 接下来就可以合理地估计出图 12.5(a)中的树的概率。这要比简单地退回到一棵未见过树的概率更可靠, 但要注意, 这里做了一个非常重要的独立性假设。例如, 假定一个 PP 附着到一个 VP(即, 在依存语法词组中一个介词依靠一个动词)的概率与这个 VP(即, 这个动词有多少个依存名词)中有多少个 NP 是无关的。我们会发现, 依存完全独立的假设的效果并不好, 而且我们也需要一些系统来说明依存关系的相对顺序。为了解决这些问题, 实际系统中采用了各种允许依存条件的方法(下面将会有相关描述)。

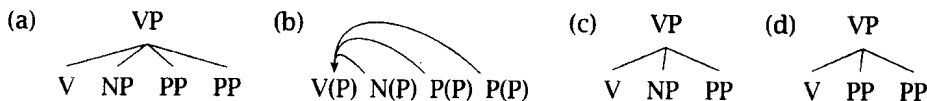


图 12.5 把局部树分解成依存关系

^① 注意, 实际上我们没有办法在依存语法内表示 2 层甚至 3 层 X' 结构, 虽然这些 X' 结构已经在现代短语结构方法中广泛使用。

12.1.8 评估

怎样评估这些统计分析器的性能是一个非常重要的问题。如果我们在构建一个语言模型(不仅仅是句法分析模型),那么可以根据留存数据计算模型的交叉熵来评估这个模型。如果我们的目标仅仅是从数据中找到一些结构形式,使我们能够更好地预测未知数据,那么交叉熵可以做得很好。但是我们最初的目的是构造一个概率句法分析器来寻找一棵合理的特定句法分析树,因此,尽管交叉熵也许可以用做评估度量,但是通过计算交叉熵来完成评估与我们的目标很不一致。交叉熵或混乱度只能衡量模型之间的概率弱等同性,但对于句法分析树来说效果并不好。特别需要指出的是,概率弱等同语法的交叉熵是相同的,但是如果它们不是强等同的,实际处理中我们可能会偏向于其中的某一个。

为什么我们对句子的特殊分析树感兴趣呢?实际上,人们很少对句法结构分析本身感兴趣。假设我们最终的任务是构造一个系统,进行信息抽取、问题回答、翻译或其他应用。原则上评估这些分析器的最好办法是把它们嵌入一个大的系统中,然后再调查不同的分析器在这种以任务为基础的评估中造成的差别。这也是处于句法分析领域之外的人们真正关心的差别。

但是,出于简单和模块化的考虑,我们常常需要使用测量方法方便地对分析器进行评估,而且这也会促使它们在任务中获得更好的性能。如果我们有很好的理由相信特定风格的分析树很有用,就可以将程序找到的分析结果与人工分析的结果进行比较,这被认为是一个黄金标准(gold standard)。但是如何去评价我们的句法分析方法呢?换句话说,什么才是我们努力去最大化的目标准则呢?最严格的标准是,如果分析器得到了完全正确的结果则取1,如果有任何错误都取0。这种标准称为分析树准确率或精确匹配准则。这是最为严格的标准,在应用中它是很有用的。其部分原因是,在很多标准的分析思想中,譬如PCFG中的Viterbi算法,就是努力最大化了这种度量。对于很多潜在的任务而言,部分正确的分析并没有多大用处,因此这是一个合理的目标准则。例如,在一个数据库查询系统中得到部分正确的分析树是没有什么用的。

另一方面,分析器的设计者,比如学生,只希望所得到的句法分析结果中大部分是正确的,甚至为了某些目的只需要部分句法分析结果正确。一般来说,分析器评估最常用的尺度是PARSEVAL度量,它最初是用于比较非统计分析器的表现的。这些测量尺度评估了分析器的各组成块。

图12.6给出了分析树、黄金标准分析树和经常应用于统计NLP工作的PARSEVAL度量结果的一个例子。通常有三个基本的衡量尺度:准确率是指在分析中有多少个括号与正确分析树是匹配的,召回率是指在分析中有多少个正确分析树中的括号,交叉括号给出了在一棵树中与其他树的成分边界交叉的成分数目的平均值。

图12.7可以帮助我们理解交叉括号(crossing bracket)的概念,交叉括号会导致非常严重的错误。与为每棵树都给出交叉括号(独立于树的大小)相比,另一种可行的方法是给出非交叉准确率(noncrossing accuracy),即指明括号不为交叉括号的百分比。最初的PARSEVAL度量忽略了节点标号、一元分支节点和执行各种其他(特别)树归一化操作(tree normalization)。但是,如果让一个句法分析器复制手工的分析结果,通常需要包括节点标号,并给出标记的准确率和

召回率的度量。一元根节点和预终结节点(preterminal node)不计在内,包括一元边界节点应该是合理的^①。

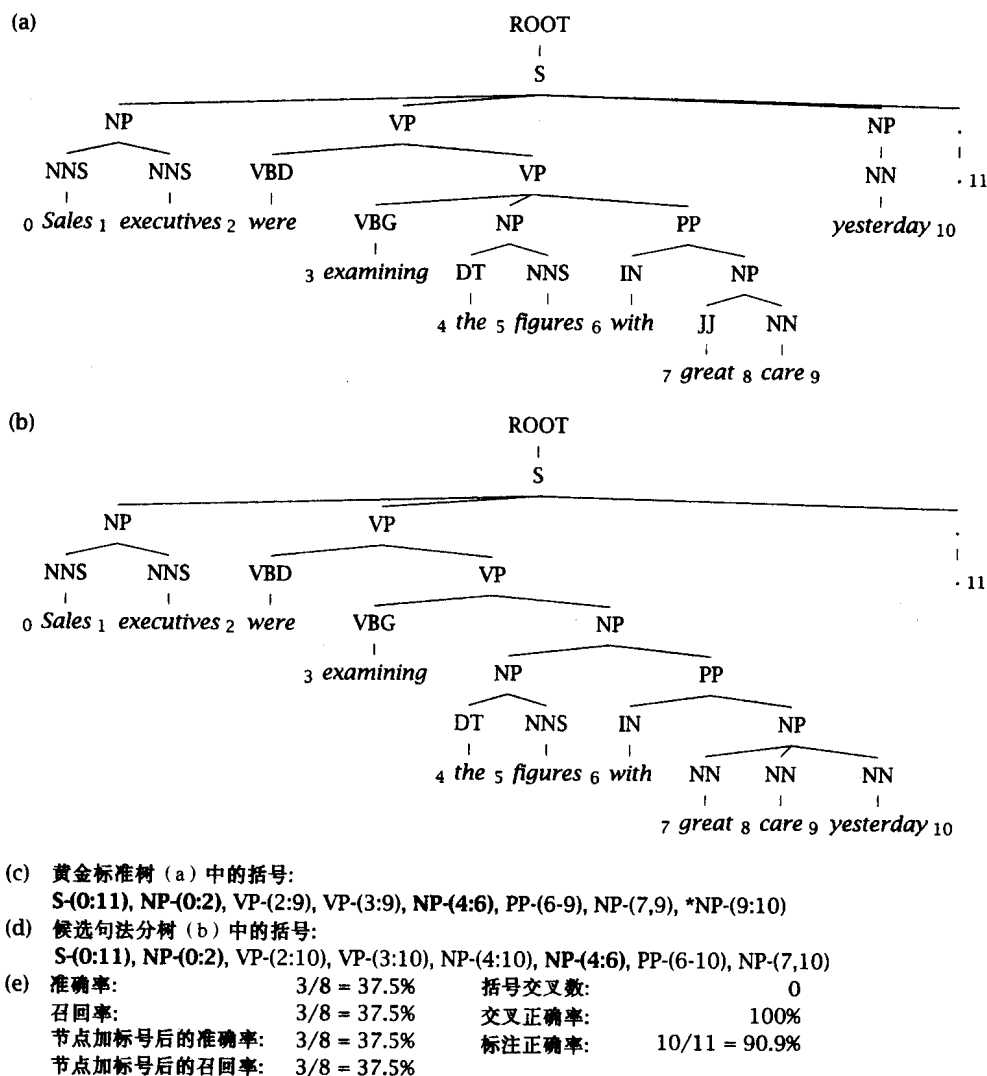


图 12.6 PARSEVAL 度量的一个例子。PARSEVAL 度量可以很容易地通过抽取非终结节点覆盖的范围来进行计算,如(c)和(d)所示,然后计算交叉,是否包括标号都可以。匹配的括号用加粗字体表示。在所有的计算中都忽略了根节点,并且预终结节点只用在标注精度计算中。根据最初的准则加星号的一元节点在计算中应该被排除掉,但是在这里被包括进来了

① Penn 树库给予了所有的树一个没有标号的顶部节点,这里称之为根节点。这使得我们可以用规则把根节点重写为句子的实际顶端节点,不论是 S 还是 NP 或是其他(在新闻文本中大量的句子实际上只是名词短语)。包含这个节点以及预终结节点会增大准确率和召回率数字,因为它们都是不会出错的一元节点。在标注度量中包括预终结节点标号的理由更加充分,但是这意味着我们做词性标注的能力取决于句法分析器的性能,然而有些人觉得最好能够分开说明。一元节点链显示出了这些度量的一些问题:这些度量无法体现节点的支配顺序,而且如果多个节点有同样的类别,在计算召回率的时候必须要谨慎。最后,我们需要注意,很多基于 Penn 树库的统计 NLP 句法分析器的评估都使用自己的特殊归一化准则,原则上没有计算句子内部的标点符号以及 ADVP 和 PRT 类别之间的差别。

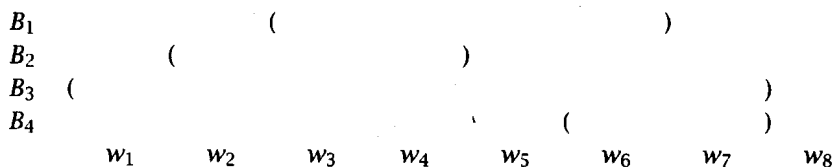


图 12.7 交叉括号的基本思想。括号 B_1 与 B_2 和 B_4 交叉。其他的所有括号都满足一致性。从直觉上讲交叉括号不能组成单个树结构

PARSEVAL 度量不是很有辨别能力。下面将会看到, Charniak (1996) 研究表明, 根据这些尺度, 利用忽略了词汇内容的 vanilla PCFG 来分析 Penn 树库的效果出奇得好。这些结果似乎反映了在许多方面根据 PARSEVAL 度量得到好的性能十分容易, 特别是对于 Penn 树库假定的这种树结构。首先, PARSEVAL 度量的成功之处在于它们的度量是基于独立判断 (individual decision) 的。一般来说, 一些自然语言处理的问题之所以很难解决, 是因为必须连续做出一系列判断才能保证问题解决成功, 整个问题的成功率是每个独立判断成功概率的乘积, 所以这个值很小。

但是, 除此以外, 还有一些针对 Penn 树库结构的大量特殊特征会使得这些度量非常容易。交叉括号的成功得益于 Penn 树库结构的平坦特性。在很大程度上讲, 树库中的句子含有相对较少的括号, 交叉括号的数量可能非常少, 因此避免了由于鉴别括号错误可能导致的准确率和召回率降低的情况。例如, 回忆一下, 在 Penn 树库中没有复合名词结构消歧, 它为下面的复合名词 (及其前置修饰语) 给出了一个完全平坦的结构 (注意, 第一个例子也阐释了 Penn 树库中把带连字符的复合名词的非终结部分标注为形容词的情况)。

[NP a/DT stock-index/JJ arbitrage/NN sell/NN program/NN]
[NP a/DT joint/JJ venture/NN advertising/NN agency/NN] (12.26)

Penn 树库的另一个有用特性是给予中心名词后置修饰语以非标准附属结构 (nonstandard adjunction structure), 并且该结构的表示采用了通用形式 (NP (NP the man) (PP in (NP the moon)))。正如我们在 8.3 节中所讨论的, 一个经常出现的歧义现象是 PP 是附着于前面的 NP 还是 VP (或更高层节点), 这就是词汇或上下文信息比结构因素更重要的一种情况。注意, 现在采用上面的附属结构减少了对做出错误决策的惩罚。图 12.8 说明了 Penn 树库类型的结构和语言学中经常假设的 N' 结构中采用的不同树括号形式。不同的附着错误如表 12.5 所示。Penn 树库结构的容错性是很明显的, 我们可以得到错误的但没有任何交叉括号的附着形式, 而且它在准确率和召回率上的错误都很小^①。

VP 附着的 Penn 树	(VP saw (NP the man) (PP with (NP a telescope)))
NP 附着的 Penn 树	(VP saw (NP (NP the man) (PP with (NP a telescope))))
VP 附着的另一类树	(VP saw (NP the (N' man)) (PP with (NP a (N' telescope))))
NP 附着的另一类树	(VP saw (NP the (N' man (PP with (NP a (N' telescope)))))

图 12.8 Penn 树和其他类型的树的对比

另一方面, PARSEVAL 度量至少有一个方面显得比较粗糙。如果一个成分附着的层次很高 (在一个复杂的右分支的句子中), 但是分析器错误地将它附着在很低的层次, 那么在这个右分支的复杂句子中的每个节点都会出错, 严重影响了准确率和召回率的计算, 尽管分析器仅

^① 这个比较假定包括了一元括号。即使不这样做, 一般的对比还是存在的, 但非 Penn 的情况的坏处会稍微减少。

仅只犯了一个错误。这就是图 12.6 碰巧给出了一个非常糟糕的结果的原因。在候选分析器中有两个附着错误,其中导致结果中有严重错误的是把 *yesterday* 附着在很低的层次时的情况(产生这个例子的分析器并不知道这个时间名词造成了巨大的损害)。

表 12.5 对不同样式短语结构的 PP 附着错误的准确率和召回率的评估结果

错误		错误评估		
			Rec.	CBs
Penn 树	VP instead of NP	0	1	0
	NP instead of VP	1	0	0
另一类树	VP instead of NP	2	2	1
	NP instead of VP	2	2	1

这些都说明 PARSEVAL 度量并不是完美的,也许有人希望用其他的方法来代替它。一种想法是考虑依存性,即测量句子中有多少个依存是正确的或错误的。但是,这样做的难点在于 Penn 树库中并没有标明这些依存信息。尽管我们可以很成功地从给出的短语结构树中导出依存关系,但是并没有任何可用的黄金标准(gold standard)。

我们重新审视一下基于任务角度来评估分析器的基本思想,正确的方法是检查 PARSEVAL 度量的成功是否也意味着在实际任务中的成功。许多小的分析错误并不影响语义阐释的任务。从(Bonnema 1996; Bonnema et al. 1997)的结果可以看出这一点。例如,在一个实验中,正确语义理解的比例是 88%,尽管分析器的准确率只有 62%。PARSEVAL 度量和基于任务的性能之间的关系在 Hermjakob and Mooney(1997)的英语转德语的翻译项目中有简短的叙述。研究发现,在一般情况下 PARSEVAL 度量和翻译质量之间有紧密的联系。到目前为止标记准确率与语义上的正确翻译之间有着最好的关联度(0.78),然而与交叉括号的弱尺度之间的关系就要差得多(0.54)。是否存在着其他的评价标准来更好地评估任务的成功与否,是否不同的准则对于不同类型的任务有着更好的性能预测,这些问题仍然有待解决。但是,现在人们一般认为这些度量方式对于比较句法分析器来说已经足够了。

12.1.9 等同模型

在比较两个概率语法的时候,很容易想到它们的区别在于使用了不同的信息处理方式,但是最基本的是它们用什么信息来进行什么预测。如果这些问题的答案是一样的,那么这些概率模型就是等同的。

通常来讲有三种不同的方法来考虑问题:记忆更多的派生历史信息,在短语结构树中查看更多的上下文,或者以确定的方式丰富树的词汇表。

我们来举个简单的例子。Johnson(1998)说明了在 PCFG 中重写一个父亲非终结节点(P)的时候使用祖父节点(G)作为附加的上下文信息是很有用的。例如,考虑分析树(12.27),



当扩展分析树(12.27)中的 NP 非终结符时,对于 NP_1 ,我们会使用: $P(NP \rightarrow \alpha | P = NP, G = S)$, 然而对于 NP_2 ,我们要使用: $P(NP \rightarrow \alpha | P = NP, G = VP)$ 。这个模型也可以体现表 12.3 所展示的主语和宾语 NP 概率分布上的差别(然而表 12.4 却没有展示出这种分布上的差别)。使用包含祖父节点的信息具有意想不到的有效性。Johnson指出,这个简单的模型实际上比前面概率左角模型的性能更好。我们知道,PCFG模型缺乏词汇化,并存在数据稀疏等方面的局限性,而该模型则可以看成是对PCFG模型的一个简单有效的改进。

但是,这里需要说明的一点是,我们可以从三条途径来考虑这个模型:使用更多的派生历史,使用更多的分析树上下文,或者丰富类别标号。第一种途径是考虑派生项,类似于基于历史的语法,在这里可以构建一个更好的派生历史等价类。对于两个相等的派生历史,这就意味着它们不仅在剩余的短语标记中含有相同的最左非终结符,而且它们必须是同类别重写的结果,即:

$$\pi(h) = \pi(h') \text{ iff } \begin{cases} \text{leftmost}_{NT}(\alpha_m) = \text{leftmost}_{NT}(\alpha'_m) = N^x \text{ \& } \\ \exists N^y : N^y \rightarrow \dots N^x \dots \in h \wedge N^y \rightarrow \dots N^x \dots \in h' \end{cases}$$

如果两个非终结符处于不同的等价类,它们一般(或者经常)会有不同的重写概率。

如果不这么做,我们可以简单地按照树结构的概率来考虑这个新模型,但是建议最好不要采用只通过查看包含子树的节点的方法得到局部子树的概率,应当考虑更多的近邻上下文。如果一次从各个方向来查看周围的上下文,也许会比较麻烦,因为这样无法得到一个有较好基础的概率模型或分析模型——使用上下文的时候只能是某个特定方向。但是,如果考虑一棵自上而下构建的分析树,那么当然可以包含尽可能多的高层上下文信息。构造派生过程序列中的等价类相当于构造部分树的等价类,而仅仅包含祖父节点的识别则是一个丰富上下文表示的简单例子而已。

第三,我们也可以像Johnson一样只采用一个普通的 PCFG 分析器,但是需要丰富树标号的词汇表,以便对额外的上下文信息进行编码。Johnson使用记录了节点的初始标号及其父节点标号的复合标号,重新标注了非终结符,例如把分析树(12.27)中的 NP_1 重新标号为 NP-S。新的树中的两个节点含有同样的标号,当且仅当在初始树中它们以及它们的父节点都含有同样的标号。接下来Johnson把标准的PCFG分析器作用在这些新的树上,以模拟在初始树中使用额外上下文信息的效果。以上三种方法都产生了等价的概率模型。但是第三种似乎是一个比较好记住的方法,因为通常编写高效程序以产生变形树比写一个新的概率分析器容易得多。

12.1.10 构造句法分析器:搜索算法

对于特定类型的概率语法,有一些高效算法能在多项式时间内找到最大概率的句法分析。这些算法工作的方式是通过保存某种形式的场景(tableau)来实现的,场景中保存了在进行自下而上风格的计算时句法分析所派生的步骤。这些场景按如下方式组织,如果两个子派生被放置在场景的同一个单元格中,那么我们知道它们都能够以同样的方式扩展成为更大的子派生或完全派生。在这种派生中,概率较低的派生总是导致较低概率的完全派生,因此它将会被丢弃。这种算法通常被称为Viterbi算法,在前面的章节中我们已经看到了一些实例。

但是,在使用更加复杂的统计语法机制的时候,却没有类似的算法。主要有两方面的原因。

首先,或许我们并不知道这些形式的场景分析算法。其次,以上的讨论假定通过缓存派生概率能够有效地找到分析概率。Viterbi 算法就是找到一棵树的最高概率派生的手段。如果我们能为每棵分析树定义一个惟一的规范派生(正如前面所讨论的),那么这个算法只允许我们找到树的最高概率分析。如果在派生和分析之间不存在一一对应的关系,就不存在找到最大概率分析树的多项式级别时间的有效算法。我们将在下面的 12.2.1 节中看到一个这样的例子。

对于这样的模型,寻找最优分析“解码问题”是一个指数级别的算法问题。不过我们需要能在一个大的搜索空间内实现求解的有效方法。如果我们以这种方式把分析问题看成是搜索问题,就能够利用 AI 领域中的一些常用搜索思路。下面将介绍最早同时也是最有名的统计 NLP 算法,即栈解码算法。

栈解码算法

栈解码算法是由 Jelinek(1969)提出的,最初用于噪声信道内的解码信息传输。但是,它也是一种遍历树结构空间的方法,这种遍历在统计 NLP 算法中经常出现。例如,一个派生分析模型给出了一个树结构的搜索空间,因为我们开始派生的第一步有多种选择,而且每一种又会导致派生的第二步的选择范围(通常不同)。AI 中有名的成本一致搜索算法(uniform-cost search algorithm)就是这样的:总是最先展开最小耗费的叶节点。

栈解码算法可以用一个优先级队列来描述,这个队列是一个能够压入元素和弹出最高级别的元素的有序列表。优先级队列可以用一个堆栈数据结构有效地实现^①。开始时优先级队列中只有一个元素,即分析器的初始状态。然后便会进入一个循环,其中每一步都会从优先级队列的顶部拿掉最高概率的元素,接着通过从第 n 步派生发展到第 $n+1$ 步派生进行扩展(通常有多种实现方法)。这些较长的派生会放在以概率排序的优先级队列的后面。这个过程将一直重复进行,直到优先级队列的顶部得到了一个完全派生。如果假定一个无限优先级队列,那么这个算法就肯定能够找到最高概率分析,因为高概率的部分派生总是在低概率的之前被扩展。也就是说,它是完全的(如果有,肯定能找到)而且是最优的(如果有几个结果,那么找到的是最优的)。如果假定有限优先级队列的大小,那么将不能找到最优的分析结果,但是这种思想对于找寻最优分析结果通常有很好的启发作用。术语“束搜索”(beam search)用来描述只保留和扩展最优部分结果的系统。束的搜索范围或者是固定大小的,或者是保留束内适合度在最优元素适合度的 α 因子之内的所有结果。

在这种方法的最简化版本中,如上所述,在从堆栈中拿掉最高概率的元素时,通过查看下面的合适的分析步骤,可以找到从第 n 步派生扩展到第 $n+1$ 步派生的所有可能途径,然后把第 $n+1$ 步的派生结果放回堆栈中。但是 Jelinek(1969)提出了一种优化(Jelinek 把它归功于 John Cocke),只采用最高概率的下一步,因此只把最高概率的第 $n+1$ 步派生与附加信息一起放回堆栈上,这些附加信息包括指向第 n 步状态和其他可能的扩展的信息。其后,如果状态从栈中弹出,不仅可以确定推进第 $n+2$ 步最高概率派生,而且检索出附加信息,再应用第二个最高概率规则,推进第二个最高概率的第 $n+1$ 步派生(或许是使用它自己的附加信息)。这种与附加信息一起工作的方法在实践中能有效地减小栈解码算法所需的搜索范围的大小。

^① 这在许多算法书中都有描述,如(Cormen et al. 1990)。

A* 搜索

成本一致搜索可能效率很低,因为它会把所有的部分派生(以宽度优先的方式)扩展一段特定的距离,而不是直接考虑它们是否可能得到一个高概率的完全派生。最优优先搜索算法却恰恰相反,依据它与一个完全解的接近程度来判断扩展哪一个派生。但是,实际上我们所需做的是在已经进行的派生步骤和尚待完成的工作的基础上,找到一个方法有效地结合它们二者,从而尽量扩展那些潜在的最高概率的派生。计算已完成步骤的概率是很容易的,而比较麻烦的是得出还未做的步骤的概率。虽然最正确的做法是选择一个最优估计,但是这意味着对其余步骤的概率估计等于或高于实际的代价。如果我们能够做到这一点,那么就表明搜索算法仍然是完备的,而且是最优的。以这种方式工作的搜索算法称为 A* 搜索。A* 搜索算法是非常有效的,因为它把句法分析器指引到那些看上去与导致完备派生最接近的部分派生。实际上,我们讲 A* 搜索是最有效的(optimally efficient),意味着没有其他的最优算法能够保证遍历更少的搜索空间。

其他方法

这里只是简单地涉及了搜索思想的一些表层的东西。读者可以在大多数 AI 书籍中找到更多的信息,例如(Russell and Norvig 1993; ch.3 ~ 4)。

但是要注意,在 Viterbi 算法不适用的一些情况下,人们经常会放弃有效训练,也不能再使用 EM 算法。但是我们可以使用其他方法,IBM 曾经尝试的一种方法是构造决策树来最大化树库的似然性(参见 12.2.2 节)。

12.1.11 几何平均值的应用

任何标准的概率方法最后都归结到概率乘积的形式。链式规则可以证明这个乘法序列的正确性,但是为了使模型可用,通常做出了很多条件独立的假设。但是,这些条件独立的假设通常是不合理的,因此错误便会被积累起来。特别地,不能将依存关系模型化就意味着一棵树的估计概率会变得很低。另外还有两个问题,一个是数据稀疏问题,即一些非常见结构的概率估计值会很低,另一个是类似于 PCFG 的有缺陷的模型,它们会偏向于给予短句相对于长句更高的概率。结果,有庞大分析树或很长的派生历史的句子会在现有的统计分析器中受到更多的惩罚。要解决这一问题,有人建议(Magerman and Marcus 1991; Carroll 1994)应该计算各种派生步骤的几何平均值(或等价的 log 概率平均值)。这种改变使得我们脱离了概率方法的范围(尽管这种假设很粗浅),从而进入了句法分析器的特别计分函数领域。这种方法有时在实践上证明是非常有效的,但是它处理的是问题的症状(symptom),而不是问题的原因。为了加速 chart 分析的速度,Caraballo and Charniak (1998)的研究工作表明,使用构造一个成分的所有规则概率的几何平均值,比简单地使用成分的概率来确定扩展哪一条边好得多——这不仅仅因为 PCFG 模型强烈趋向于给予一棵小规模树很高的概率,而且因为这种度量忽略了剩余树的概率。但是,如果设计出了更好的适合度概率度量(probabilistic metrics of goodness),我们就可以做得更好。

12.2 一些方法

在本章余下的部分,我们讨论的重点是如何将前面阐释的一些基本思想结合到具体的统计句法分析器中,讨论的目的是希望读者对现有研究方法的概貌有一个大致认识,所以这里给出的介绍也相对简短一些。

12.2.1 非词汇化树库语法

概率分析器中基本上可以分为词汇化分析器和那些操作在词语类别上的分析器。我们首先描述非词汇化分析器。对于一个非词汇化分析器,输入的分析句子实际上只是一个词语类别标记的列表,一个普通分析树的预终结符(preterminal)。很显然这比一个含有实际词的句子给予了更少的信息,接下来我们将讨论高性能词汇化分析器。但是,除了一般的理论上的兴趣,非词汇化分析器的好处在于终结符字母表很小,因此句法分析器的构建很容易,同时不必过分地担心计算效率或平滑稀疏数据的问题。

从树库中进行PCFG估计:Charniak(1996)

Charniak(1996)提出了一个重要实验问题:如果忽略了词汇信息,那么句法分析器的性能会如何变化。Charniak采用了Penn树库,并且采用了它的词性和短语类别标记(忽略了功能标记),通过使用局部树的相对频率显式地进行规则估计,从而从分析树中导出了一个最大似然PCFG,但是算法缺乏相应的规则平滑操作,并且试图分析以前没有出现过的句子结构^①。

结果是这种语法表现得非常好。它在准确率、召回率和交叉括号方面的性能并不比最好的词汇化分析器低很多(参见表12.6)。其实这个结果是很令人惊奇的,因为这样的分析器在处理相同结构的上下文时总是选择同样的附着歧义解决方案,因此经常是错的(参见8.3节)。可能的原因是这些计分尺度对于Penn树库分析树没有辨别能力,正如我们在12.1.8节中所讨论的。但是它或许也同样表明了分析决策,譬如经典的附着歧义,需要语义或词汇信息,但是或许分析决策的主要部分是很普通的,因此能够被一个非词汇化PCFG处理得很好。准确率、召回率和交叉括号尺度记录了平均性能,因此一般我们只用一个PCFG就能做得很好。

令人感兴趣的另一点是无需平滑导出语法就可以得到这个结果。尽管众所周知,树库中存在很多平坦多树分支成分,而实际上其中许多结构是很少见的。正如Charniak所言,从Penn树库导出的语法对句子中下一个出现的词性几乎没有直接限制,因此我们可以分析任何句子。尽管在测试集中可能出现未知的局部树,但即使是做了平滑,它们不太可能出现在高概率的分析中。因此,在这种条件下,仅仅用最大似然估计不会得到不良后果。

部分无监督学习:Pereira and Schabes(1992)

我们知道PCFG的实际参数估计空间很大,甚至连EM算法也无能为力,因为它总是陷入局部最大值。Pereira and Schabes(1992)以及Schabes et al.(1993)提出了一种改进方法,即尽可

^① 我们稍微做了一点简化。Charniak还做了两个处理,通过AUX标记记录助动词,并结合一个“右分支校正”,以便分析器偏向于右分支结构。

能事先确定一个好的参数空间。首先从Chomsky正则文法开始,该语法由15个非终结符以及包含45个终结符的字母表组成,在树库的句子训练,训练过程忽略了非终结符标记,但是使用树库的括号。另外,采用了一系列受限的Inside-Outside变形算法来分析那些在Penn树库节点中没有覆盖的树节点。其句法分析器经常分析成二元成分(binary constituent),但是能够学习任意风格的括号,把它们作为句子的部分括号(partial bracketing)。我们不再详细介绍Inside-Outside算法的变形版本,但是其基本思想是将与树库括号不一致成分的重估等式的贡献减小到零。因为使用括号减小了值得考虑的规则分割点的数目,所以一个括号训练语料库也会改善Inside-Outside算法的效率。

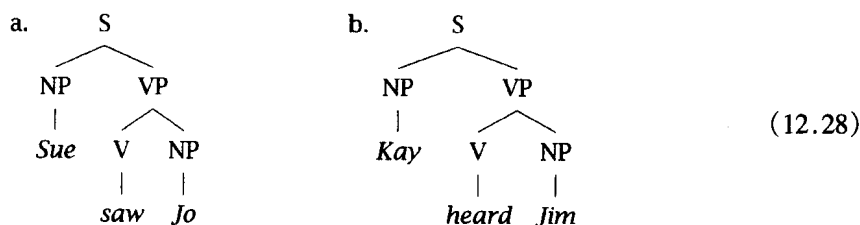
在一个小的测试集上,Pereira and Schabes(1992)说明了这个基本思想的有效性。有趣的是,在没有括号和有括号的训练数据上训练得到的语法都收敛到一个非常相似的交叉熵上,但是它们的括号与树库中出现的期望括号的对应程度有很大的出入。如果训练输入是没有括号的,那么分析器用在测试句子上只有37%的括号是正确的,但是当它经过了在已经加括号的句子上的训练,那么用在这些测试句子上的90%的括号都是正确的。而且,尽管用EM算法来训练这些没有加括号的数据在减小交叉熵方面很成功,但是这对于在随机初始化参数模型的基础上提高括号的准确率是无效的。这个结果强调了本章开始的讨论:现有的学习方法对于找到低熵的模型是有效的,但是对于从生文本中学习语法结构还不充分。只有偶然推导出的语法才会与对句子结构的通常判断一致。如今,假定的语言层次结构通常不能由生语料决定,或者它的证据过分微妙,以至于当前的归纳技术不能发现,这些仍然是有待解决的问题。

Schabes et al.(1993)在一个含有长句子的大语料库上测试了同样的方法,并且取得了相似的结果。其中利用了另外一些有趣的思想,把一致的右分支二元结构强加在训练数据中的所有平坦的 n 元分支Penn树库局部树上,以便最大限度地提高加括号的Inside-Outside算法的速度。

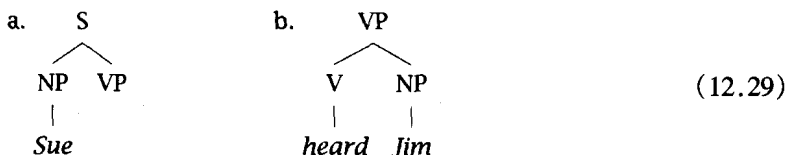
直接从树中分析:面向数据的分析

到目前为止我们考虑基于语法的模型的另一种方法是,直接从树库中树的片断来得到统计数据,其中树库假定代表了以前曾经经历过的分析。相比于从树库中派生出语法,我们趋向于在分析过程使用那些有用的分析树的片断。这样做有明显的优点,类似于*to take advantage of*的成语语块一般都会整体出现,然而这些语块并不直接被PCFG风格的模型所体现。Rens Bod and Remko Scha(Sima'an et al.1994; Bod 1995,1996,1998)的面向数据分析(DOP)的框架已经尝试了这种方法。在这一节中,我们将查看DOP1模型。

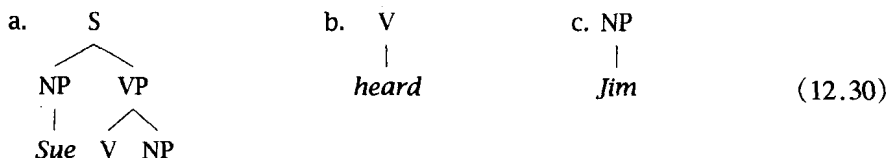
假设我们有一个含有两个句子的语料库,如分析树(12.28)所示:



于是,为了分析一个类似于 *Sue heard Jim* 的新句子,可以把曾经见过的树片断放在一起。例如,我们可以组成两个树片断:



我们可以得到语料集中每个树片断的概率。给定某一个正在展开的特定节点,同时假定存在独立性,我们就可以把这些概率相乘。例如,这里有以 VP 作为父节点的 8 个片断,片断必须包括一个节点的所有子节点或者没有子节点的节点,其中分析树(12.29b)出现了一次,因此它的概率是 1/8。但是,这仅仅只是这个分析树的一个派生。在通常情况下应该有很多派生,下面是语料库中的另外一个例子,包括了三个树片断:



既然在这个 DOP 模型中一棵独立的树有多个根本不同的派生,这种情况下最高概率的分析不一定能被 Viterbi 算法(Sima'an 1996)有效地找到,参见习题 12.8。因此,分析时采用 Monte Carlo 模拟方法。这是一种通过随机选取样本来估计事件概率的技术。我们会随机生成大量的派生,同时用它们来估计最可能的分析。有了足够大的样本集合,估计就可以做到尽可能地精确,但是分析过程将会变得很慢。

DOP 方法在从语料库中直接进行预测方面与基于记忆的学习 (MBL) 方法 (Zavrel and Daelemans 1997) 有些类似,但是不同点在于 MBL 方法的预测基于一些相似的例子, DOP 则基于对整个语料库的统计。

DOP 模型提供了一条不同的思考途径,但实际上这与我们处理 PCFG 的方法差别并不大。毕竟,相比于写出类似于 $S \rightarrow NP VP$ 和 $VP \rightarrow V NP$ 的语法规则,我们更愿意作为其替代写出树片断。



而且,我们从树库中估计得到语法规则的概率与 DOP 模型在树库中基于它们的相对频率来分配概率是一样的。

PCFG 和我们这里所做的工作之间的区别是,与只有一层深度的局部树相比,我们有更大的树片断。这个模型可以被形式化为概率树替换语法 (PTSG), 它类似于第 11 章中介绍的 PCFG 定义, 同样有 5 个组成部分。但是,与 PCFG 中的规则集合不同,我们采用了任意深度的树片断集合,它们的顶部和内部节点都是非终结符,叶节点是终结符或非终结符,而且概率函数为这些片断分配了概率。因此 PTSG 是 PCFG 的泛化,而且更加强大,因为人们可以给片断或者整个分析以特殊概率,而这些不能由 PCFG 中的规则概率相乘得到。Bod (1995) 表明,如果以一层片断的 PCFG 模型开始,然后逐渐地允许更大的片断,分析准确率将会有很大提高。这反映

了Johnson(1998)在树中利用高层节点上下文的结果。因此,DOP模型提供了另一种使用更多的上下文条件来构造概率模型的方法。

12.2.2 使用派生历史的词汇化模型

基于历史的语法(HBG)

基于派生历史的概率方法,并拥有词汇和其他信息的充分支持,IBM首先在一个大规模的实验中尝试了这种方法,(Black et al. 1993)说明了相关的情况。这项工作利用了最左派生和分析树之间的一一对应关系来避免对可能的派生进行求和。常见的思想是派生中前面的所有分析决策都会影响后面的分析决策,但是,在 1993 年的模型中,惟一考虑的条件特征是从当前节点扩展到根节点的派生路径,以及一个节点是父节点的第几个子节点(从左至右)^①。Black et al. (1993)使用决策树来决定派生历史中的哪个特征对于确定当前节点的扩展是重要的。我们将在 16.1 节中介绍决策树,它们可以被看成只是一个把历史划分成高预测性的等价类的工具。

和其他工作不同,这项工作采用了一个由Lancaster大学标注的树库。在 1993 年的实验中,他们严格限制在语料库中的句子由 3000 个最常用词构造而成(这样可以有效地避免许多数据稀疏问题)。Black et al. 以一个手工建造、覆盖面广的基于特征的合一语法开始。通过从特定的标记中(忽略或者聚合某些特征和特征值对)构造等价类的方法来转化成一个 PCFG。然后使用一个避免括号交叉的 Inside-Outside 算法来重新估计这个 PCFG,参见上面所讨论的 Pereira and Schabes(1992)。

Black et al. 词汇化了他们的语法,因此短语节点继承了两个词,词汇中心词 H_1 和第二中心词 H_2 。词汇中心词是我们熟知的短语中的中心词,然而第二中心词是我们认为有用的另一个词(例如,在一个介词短语中,词汇中心词是介词,第二中心词是名词短语补充成分中的中心词)。而且,他们定义了一个大约有 50 个语法和语义类别的集合, $\{Syn_p\}$ 和 $\{Sem_p\}$, 用来分类非终结节点。在 HBG 分析器中,这两种特征,两个词汇中心词和将要被应用在一个节点上的规则 R 是基于父节点上同样的特征预测的,索引 I 表示了父节点的第几个子节点正在被扩展。即,我们希望计算:

$$P(Syn, Sem, R, H_1, H_2 | Syn_p, Sem_p, R_p, I_p, H_{1p}, H_{2p})$$

联合概率通过链式规则进行分解,每个特征都使用决策树单独估计。

指导 IBM 工作的思想是,与其让一个语言学家修补语法来提高分析特性,还不如让语言学家生成一个能分析所有句子的分析器。于是我们可以得到一个统计分析器来学习树库中的信息,以便它能够把分析步骤建立在派生历史上来预测正确的分析。HBG 分析器在 7 至 17 个词长度的句子上进行测试,比较了它和现有的合一分析器的性能。合一分析器有 60% 的情况下选择了句子的正确分析,然而 HBG 大约是 75%,统计分析器相比于 IBM 语言学家手工得到的最优消歧规则减少了 37% 的错误率,因此是很成功的。

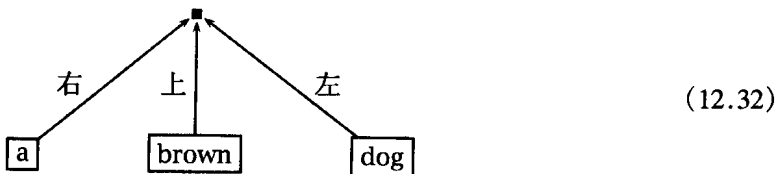
^① 简单地使用父节点的第 n 个子节点作为特征,在语言学上似乎有些不可靠,因为不知道其他的子节点是什么样的,但是这对表 12.4 中所展示的变化分布给出了一些可操作的手段。

SPATTER

HBG是基于语言模型的,但是IBM的工作是以直接构造一个分析模型来开始实验的。在Jelinek et al.(1994)的报告中提到了Magerman(1994, 1995)有关SPATTER模型的早期研究工作,下面我们将会进行简要介绍。

SPATTER也是通过确定派生的概率来工作的,但是它是自下而上的,以词语开始,然后在它们的基础上构造结构。决策树模型又被用于从派生历史中取出特征来预测特定的分析决策。SPATTER开始趋向于把局部短语结构树分解成单独的分析决策,但它不像其他工作那样使用依存语法,而是使用了一些奇怪的技术来预测一个节点上的分支指向。

在SPATTER中,一棵分析树是按照词、词性标记、非终结符标记和树形式的扩展来编码的。词性标注是分析过程的一部分。既然语法已经被充分地词汇化,中心子节点的标记和词通常会追溯到非终结符节点。如果我们以一些词开始并且希望预测它们形成的子树,就会类似下面的情况:



预测一个节点的扩展,即找出它上面联系着父节点的连线的类型。有五种扩展:对于有两个或更多分支的子树,右(right)被分配给最左子节点,而左(left)被分配给最右子节点,上(up)被分配给左右之间的子节点,一元(unary)被分配给惟一子节点,根(root)被分配给树的根节点。注意右和左是交换过的!

这些特征,包括词语的词性标注,都通过决策树模型来预测。对于一个节点而言,特征是根据它周围的和低层节点的特征来预测的,它们事先已被确定。模型回答如下问题(X 是上面提到的四个特征之一):

- 当前节点或节点{1或2}是{左或右},那么 X 是什么?
- 在当前节点的{第一或第二}{最左或最右}子节点的 X 是什么?
- 该节点有多少个子节点?
- 词语的节点间的跨距是多少?
- 对于标记,前面的两个词性标记是什么?

句法分析器允许遍历不同的派生序列,因此它可以从能够获得最好的预测信息的地方开始工作(尽管实际上可能的派生顺序是有限制的)。句法分析概率通过对派生求和得到。

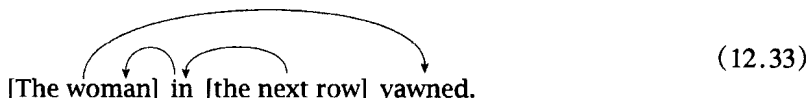
SPATTER的一些特征,比如扩展特征,是很奇怪的,而且总体结果是一个需要强大计算能力来训练和运行的大而复杂的系统(决策树训练和平滑算法的计算量都特别大)。但是它的成功是不容置疑的。SPATTER表明我们能够自动地从树库数据中导出一个成功的统计句法分析器,并且它在处理任意文本方面比任何目前存在的手工句法分析器都更好。

12.2.3 基于依存关系的模型

Collins(1996)

最近Collins(1996, 1997)从树库数据中得到了一个概率分析模型,它比前面章节中提到的模型更简单、更直观,计算速度更快,但是性能与其一样或更好。

Collins(1996)引入了一个词汇化的类似于通用依存语法的框架,而Penn树库中的baseNP单元是作为语块对待的。在分析中以这种方式使用语块可以追溯到Abney(1991)的方法。初始的模型也是一个分析模型。一个句子由一些基本NP和其他的有依存关系(D)的词(B)来表示。



于是:

$$P(t|s) = P(B, D|s) = P(B|s) \times P(D|s, B)$$

标注是一个独立的过程,通过Ratnaparkhi(1996)的最大熵标注器来实现。baseNP的概率估计采用了Church(1988)来识别NP(参见10.6.2节)的思路。词语之间的每一个间隔 G_i 都被分为NP的开头,NP的结尾,两个NP之间或者三者都不是。对于一个从 w_u 开始长度为 m 的baseNP β 的概率,根据预测的间隔特征给出:

$$P(\beta|s) = \prod_{i=u+1}^{u+m} \hat{P}(G_i | w_{i-1}, t_{i-1}, w_i, t_i, c_i)$$

c_i 代表了在词语之间是否有一个逗号。删除插值用来平滑这个概率。

对于依存模型,Collins用中心词替换了每个baseNP,而且删掉了标点,以给出一个简化的句子。但是标点是用来指导分析的。Collins的方法中比较巧妙的部分是直接采用了Penn树库中的短语结构,但是自动从它们中间派生出了依存关系符号。依存性是基于中心词和两个子节点成分命名的。因此如果有一棵如分析树(12.34)的子树,那么PP和动词之间的依存关系可以描述为VBD_PP。



换句话说,依存名称是从纯粹的类别标记中派生出来的,但是它具有我们将会在句法分析器中用到的很多功能信息。尽管如此,系统仍然还有一些局限性。例如,这些依存关系在双宾语动词的两个宾语之间并没有标明差别。

每个依存关系都假定是独立的——一种不太实际的假定。于是,除了句子的主要谓词之外的每一个词 w_m ,都会通过一种依存关系 $R_{w_m, h_{w_m}}$ 依赖于某个中心词 h_{w_m} 。因此 D 可以写成一系列依存关系 $\{d(w_i, h_{w_i}, R_{w_i, h_{w_i}})\}$,我们有:

$$P(D|S, B) = \prod_{j=1}^n P(d(w_j, h_{w_j}, R_{w_j, h_{w_j}}))$$

Collins 计算了两个词语-标记对 $\langle w_i, t_i \rangle$ 和 $\langle w_j, t_j \rangle$ 在 Penn 树库中显式地以关系 R 出现在同样的简化句子中的概率。他计算了某一种关系相对于整个关系空间的频度:

$$\hat{F}(R | \langle w_i, t_i \rangle, \langle w_j, t_j \rangle) = \frac{C(R, \langle w_i, t_i \rangle, \langle w_j, t_j \rangle)}{C(\langle w_i, t_i \rangle, \langle w_j, t_j \rangle)}$$

然后他把这个度量进行归一化,以给出一个概率。

这个模型也因为增加了基于依存关系拉伸的距离的条件而变得很复杂,这里的距离由一个不仅包括了距离,而且还包括了方向、中间是否有动词以及中间有多少个逗号等特殊函数衡量。

这个分析器采用了带有各种剪枝启发的束搜索法 (beam search)。整个系统可以在 15 min 内训练完成,而且运行得很快,即使只使用一个很小的束,系统性能也很好。Collins 的分析器比 SPATTER 性能要稍微好一点,但主要的进步似乎在于构造了一个简单快速并且性能还可以接受的系统。Collins 同时采用了使用和不使用词汇信息的两种方法来评估他的系统,表明词汇信息在标注精度和召回率上有大约 10% 的提高。比较奇怪的是,他的结果中的非词汇化版本比 Charniak 的 PCFG 分析器表现得差得多。我们可以假设,虽然在处理词汇化模型的时候把局部子树分解成独立的依存关系对于避免数据稀疏问题很有用,但是这意味着我们没有抓住在基本 PCFG 模型中的一些有益(统计)依存关系。

词汇化基于依存关系的语言模型

Collins (1997) 在其 1996 年的工作基础上重新开发了一个生成语言模型(初始的是一个有缺陷的概率分析模型)。Collins 构造了一系列复杂的逐步递增的模型,每一步性能都会有所提高。语言模型的一般方法是以一个父节点和中心词开始,然后将对中心词左右的词连续生成依存关系的过程模型化。在第一个模型中,每一个依存的概率独立于其他的依存(它依赖于父节点和中心节点的类别,中心词条和一个由距离函数、中间词和标点组合而成的复合特征函数)。依存将会持续生成,直到生成一个特殊的伪非终结符 STOP。

接下来,Collins 努力构造一个更加复杂的模型,使其能够抓住中心词的不同依存之间的(统计)依存关系。特别有趣的是,这个模型开始引入了很多传统的语言学。第二个模型利用了论元和附属修饰成分的差别,并且将中心词的子类化框架模型化了。对于每个中心词,都预测了一个子类化框架,而且依存的生成是额外建立在一些还没有生成的子类化论元之上的。试图模型化子类导致的一个问题是,各种子类化论元也许不会出现在它们通常的位置上,因为有类似隐藏论元问题(见 8.4 节)或 *wh* 移动的过程。在最终的模型中,Collins 准备把 *wh* 移动包含进概率模型中,通过利用跟踪和联合索引的填充器(在 Penn 树库中出现过)来实现。虽然第二个模型比第一个的性能好,但终究第二个系统是复杂的,看上去并没有给出更好的性能。

12.2.4 讨论

表 12.6 中列出了一些可以大致比较的系统的整体分析性能参数^①。在写这本书的时候,

① 所有这些系统都是在 Penn 树库上进行训练的,并且在一个未知的含有 2 个至 40 个词的句子集合上进行过测试,这个集合也来自 Penn 树库。但是,各种系统在标点符号的处理和选择忽略某些非终结符差别等方面是不同的,意味着这些结果并不是可以精确比较的。SPATTER 的结果是 Collins (1996) 将 SPATTER 运行在他自己的句法分析器所采用的测试数据上时给出的结果,与 (Magerman 1995) 所给出的结果有些不同。

Collins的结果在众多的统计分析器中是最好的。是否能够把IBM的工作中有用的部分与Collins工作的关键思想衔接好(例如,利用留存数据估计模型参数、决策树的应用以及更加精巧的删除估计技术),以生成更好的分析器,仍然是一个有待解决的研究问题。另外,我们注意到有另外几个系统有着差不多好的性能,它们使用了十分不同的分析技术,因此看上去对于其他的技术仍然有广阔的探索空间。例如,Charniak(1997a)对于常用的语法规则(适当的词汇化)使用了概率估计。扩展节点的规则是基于节点类别、父节点类别和它的中心词预测的。每个子节点的中心词是基于子节点类别、其父节点类别和它的中心词预测的。对于使用最近几个流行的统计分析器的条件的差别,以及什么是决定性能好坏的主要支配因素,Charniak进行了特别深入的分析。

表 12.6 一些统计句法分析器系统的比较。LR = 标注召回率, LP = 标注准确率, CB = 交叉括号, n/a表明结果没有给出。Charniak(1996)给出的非交叉括号准确率为 87.7%

	少于40个词的句子			
	% LR	% LP	CB	% 0 CBs
Charniak (1996) PCFG	80.4	78.8	n/a	n/a
Magerman (1995) SPATTER	84.6	84.9	1.26	56.6
Collins (1996) best	85.8	86.3	1.14	59.9
Charniak (1997a) best	87.5	87.4	1.00	62.1
Collins (1997) best	88.1	88.6	0.91	66.5

正如在标注中一样,可以获得的丰富词汇源(一般情况下为Penn树库)和统计技术的使用,为句法分析性能带来了新的水平。但是我们注意到,最近的一些进步虽然很重大,但还是没有突破性进展。正如Charniak(1997a:601)指出的:

这似乎表明,如果我们的目标是得到 95% 的平均标注准确率和召回率,那么在目前使用的基本框架下,即使有更长足的进步也不能让我们实现这个目标。

质的飞跃也许需要语义上更加丰富的词汇源和概率模型。

12.3 深入阅读

读者可以在两年举办一次的 International Colloquium on Grammar Inference (Carrasco and Oncina 1994; Miclet and de la Higuera 1996; Honavar and Slutzki 1998)中参阅大量的语法归纳方面的工作。

非严格限制文本上的概率句法分析的当前工作出现在 DARPA Speech and Natural Language community 中。通常被引用的论文包括 (Chitrao and Grishman 1990) 和 (Magerman and Marcus 1991)。特别地, Magerman 和 Marcus 最早涉及到了不同位置 NP 的可变结构属性。

统计句法分析早期工作的另一条线索出现在 Lancaster 大学。Atwell(1987)以及 Garside and Leech(1987)描述了一个类似于 (Church 1988) 中的 NP 查找的成分边界查找器。在一个小的树库上训练出的 PCFG 被用来在可能的成分之间进行选择。另外也出现了一些有关使用模拟退火的概率的讨论。据称他们的系统能够在 50% 的情况下找到可以接受的句法分析。

另一个统计句法分析工作的竞技场是在模式识别领域中,由King-Sun Fu领导,详情参见(Fu 1974)。

包括了词性标注的统计句法分析的方法性介绍在(Charniak 1997b)中。(Marcus et al. 1993)和(Marcus et al. 1994)中讨论了Penn树库的设计,该设计可以从Linguistic Data Consortium中获得。

最初的PARSEVAL度量出现在(Black et al. 1991)和(Harrison et al. 1991)中。对于各种句法分析评估度量,它们之间的关系,以及对不同目标函数的合适的分析算法的研究可在(Goodman 1996)中找到。

依存语法的思想可以延伸到中世纪阿拉伯语法学家的工作,但是在Tesnière(1959)的工作中得到了明确的形式化描述。在概率依存语法方面,最初的工作是Lafferty et al. (1992)的概率连接语法模型。除了一个词可以双向连接的奇特属性,连接语法可以看成是依存语法的一个符号变换。其他的基于依存的统计分析器方面的工作包括Carroll and Charniak(1992)。

我们仅仅只讨论了当前统计句法分析的大量工作中的少量论文。与(Collins 1997)性能非常类似,但是使用了非常不同的语法模型的系统,曾经由Charniak(1997a)和Ratnaparkhi(1997a)列出。也可以看一看(Eisner 1996)中基于依存的统计模型的另一种最新的方法,它非常类似于(Collins 1997)。

这里的大部分工作都是以上下文无关文法为基础的概率句法分析。另外有一些更强大的语法框架的概率版本方面的工作。Resnik(1992)和Schabes(1992)讨论了概率TAG(邻接树语法)。在早期类似中心词驱动的短语结构语法(Pollard and Sag 1994)和词汇功能语法(Kaplan and Bresnan 1982)的概率合一语法工作中,例如(Brew 1995),使用了不合适分布,因为合一语法内部的依存关系没有被合适地说明。这项工作的一个比较坚实的基础是由Abney(1997)提供的。参见(Smith and Cleary 1997)。Bod et al. (1996)以及 Bod and Kaplan(1998)在LFG上尝试了DOP方法。

基于转换的学习方法已经被应用在句法分析和语法归纳上(Brill 1993a, c; Brill and Resnik 1994)。参见第10章,其中对基于转换学习的方法有一般性的介绍。

Hermjakob and Mooney(1997)把一种基于机器学习的技术(决策列表)的非概率句法分析器应用在树库句法分析的问题上,而且获得了很好的结果。他们的工作中值得继承的对今后的统计NLP研究有用的信息是从语义类别特征中得到的,然而大部分现今的统计NLP工作都趋向于过分强调语法特征(显然是因为语法特征是从当前可获得的树库中最容易获得的)。

Chelba and Jelinek(1998)最先展示了一种性能优于三元语法模型的概率句法分析器作为语言模型用于语音识别。他们使用了一种词汇化二元语法(基本等同于依存语法),而且依据较大成分中还没有被包含的两个前面的中心词来预测词语。

本章中的大部分例子都把句法分析本身当成了一个目标。这是因为句法分析器还没有表现得足够好,以及句法分析还没有被应用于类似于语音识别和语言理解的更高层次的任务。但是,在语义理解上已经引发了很大的兴趣,一种方法试图在一个整合了语法和语义处理的过程中从句法分析构造一个句子的语意表达。参见(Ng and Zelle 1997)和最近的概述文章。Miller et al. (1996)中描述了在统计上用各种方法从词语到词语表达来训练处理句子并用于一个航空订票系统。

12.4 习题

12.1 [★]本章开始时参考的华尔街日报中的文章的第二句是:

The agency sees widespread use of the codes as a way of handling the rapidly growing mail volume and controlling labor costs. (12.35)

在这个句子中找出至少五个完整的句法结构。如果做不出这个习题,可以尝试先做习题 12.2。

12.2 [★★]写出一个有重写规则的上下文无关文法的句法分析器,并且使用它来找到句子的所有分析。使用这个句法分析器和式(12.36)中的语法来分析习题 12.1 中的句子。因为没有用习惯的简写语法格式,所以它是冗长而难看的,例如没有使用通常用在短语结构语法上的分支选择。另一方面,可以很容易地写出处理这种形式的语法的句法分析器。你得到了多少分析? 答案应该是 83。

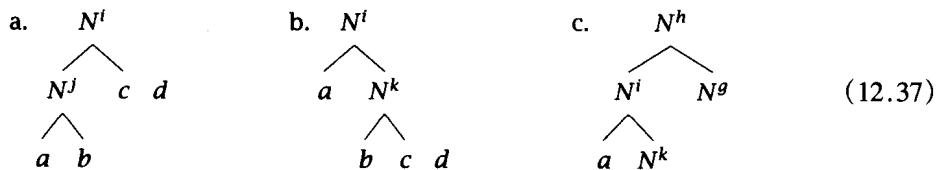
- a. $S \rightarrow NP VP$
- b. $VP \rightarrow \{ VBZ NP \mid VBZ NP PP \mid VBZ NP PP PP \}$
- c. $VPC \rightarrow VBG NP$
- d. $NP \rightarrow \{ NP CC NP \mid DT NBAR \mid NBAR \}$
- e. $NBAR \rightarrow \{ AP NBAR \mid NBAR PP \mid VPG \mid N \mid NN \}$
- f. $PP \rightarrow P NP$
- g. $AP \rightarrow \{ A \mid RB A \}$
- h. $N \rightarrow \{ \text{agency, use, codes, way, mail, volume, labor, costs} \}$ (12.36)
- i. $DT \rightarrow \{ \text{the, a} \}$
- j. $V \rightarrow \text{sees}$
- k. $A \rightarrow \{ \text{widespread, growing} \}$
- l. $P \rightarrow \{ \text{of, as} \}$
- m. $VBG \rightarrow \{ \text{handling, controlling} \}$
- n. $RB \rightarrow \text{rapidly}$
- o. $CC \rightarrow \text{and}$

在写句法分析器的时候,把规则与概率联系起来,就可以使用这个句法分析器做本章后面讨论的类型的实验。

12.3 [★]在第 11 章中,我们表明 PCFG 对使用较少扩展的非终结符有坏的偏置。假设我们有下面给定的树库作为训练集,其中“ $n \times$ ”指明了一个特定的树在训练集中出现的次数。我们可以从树库中得到什么样的 PCFG 呢(使用文中讨论的 MLE)? 如果使用这个语法,那么字符串‘aa’最可能的分析是什么呢,这是一个合理的结果吗? 第 11 章中提出的偏置问题是正确的吗? 请讨论。

$$\left\{ \begin{array}{ccccc} \begin{array}{c} S \\ \wedge \\ 10 \times \begin{array}{c} B \ B \\ | \ | \\ a \ a \end{array} \end{array} & \begin{array}{c} S \\ \wedge \\ 95 \times \begin{array}{c} A \ A \\ | \ | \\ a \ a \end{array} \end{array} & \begin{array}{c} S \\ \wedge \\ 325 \times \begin{array}{c} A \ A \\ | \ | \\ f \ g \end{array} \end{array} & \begin{array}{c} S \\ \wedge \\ 8 \times \begin{array}{c} A \ A \\ | \ | \\ f \ a \end{array} \end{array} & \begin{array}{c} S \\ \wedge \\ 428 \times \begin{array}{c} A \ A \\ | \ | \\ g \ f \end{array} \end{array} \end{array} \right\}$$

- 12.4 [★★]我们可以把 CFG 最左派生和一个 n 元语法模型结合起来产生一个使用短语结构的概率语言模型吗? 如果可以, 我们必须做出什么样的独立性假设呢? (如果这种方法看上去很有意思, 你可以尝试去实现它!)
- 12.5 [★]PLCG 对于 NP 在主语位置和宾语位置的特定扩展可以有不同的概率, 我们注意到, 在脚注中提到 PLCG 并不能抓住 NP 作为动词的第一个或者第二个宾语时的不同分布, 如表 12.4 所示。解释一下为什么会这样。
- 12.6 [★★★]如 Demers(1977)中所示, 左角句法分析器、自上而下的句法分析器和自下而上的句法分析器都适合于一个广义左角句法分析器系列, 它们的行为依赖于在进行各种动作之前查看了多少输入元素。这就表明了其他可能的概率句法分析模型实现了这个空间的其他一些点。在这个空间中还有其他有用的点吗? 对于这些点来说什么是合适的概率模型?
- 12.7 [★★]在 12.2.1 节中我们指出, 一个非词汇化句法分析器总是在同样的结构配置中选择同样的附着。但是, 考虑 8.3 节中讨论的 PP 附着问题, 这并不意味着它必须总是为 PP 选择一个名词附着或动词附着。为什么呢? 在语料库中调查是否有可利用的东西能够区别 PCFG 能区别的情况。
- 12.8 [★]这个习题的目标是让我们意识到为什么不能为 DOP 句法分析构造一个 Viterbi 算法。对于 PCFG 句法分析, 如果我们已经构造了分析树(12.37a)和分析树(12.37b)中展示的两个成分(或部分)派生, 而且分析树(12.37a)中的 $P(N^i)$ 比分析树(12.37b)中的大, 就可以丢弃分析树(12.37b), 因为用分析树(12.37b)构造的比较大的树比同样的树由分析树(12.37a)构造的概率更低。但是我们不能在 DOP 模型中这么做。为什么不能呢? 提示: 假定树的碎片(12.37c)也在语料库中。



- 12.9 [★★★]使用 Penn 树库构造、训练和测试你自己的统计句法分析器。如果你选择尝试一些很清楚的假设, 那么你的结果对别人可能是很有用的。

第四部分 应用与技术

第 13 章 统计对齐和机器翻译

第 14 章 聚类

第 15 章 信息检索

第 16 章 文本分类

第 13 章 统计对齐和机器翻译

机器翻译(machine translation),顾名思义,就是自动地将文本或者谈话内容从一种语言翻译为另外一种语言,属于自然语言处理的最重要的应用领域之一。作为一个自然语言处理领域的研究者,减少来自不同地域、不同文化背景的人们的交流障碍,一直是我们的梦想。

然而遗憾的是,机器翻译任务本身是一个很难解决的问题。现在我们可以市场上购买到许多种类的机器翻译软件,尽管价格很便宜,但是由它们翻译得到的经常是质量低劣的译文,还需要用户的重新编辑,对于精通外语的专业人员或许它是合适的,但对于普通的用户来说却令人难以忍受。自然语言处理的最终目标就是生成没有任何错误的译文,让普通人也能流畅地阅读。当然,在一些受限领域,例如天气预报(Isabelle 1987),系统的性能有可能接近实用。但是,现有的绝大多数系统距离这一目标都非常遥远。

机器翻译的困难来自何处?回答这个问题的最好方法是首先考察一下现有的机器翻译中所采用的各种方法,图 13.1 列出了其中最重要的几种。

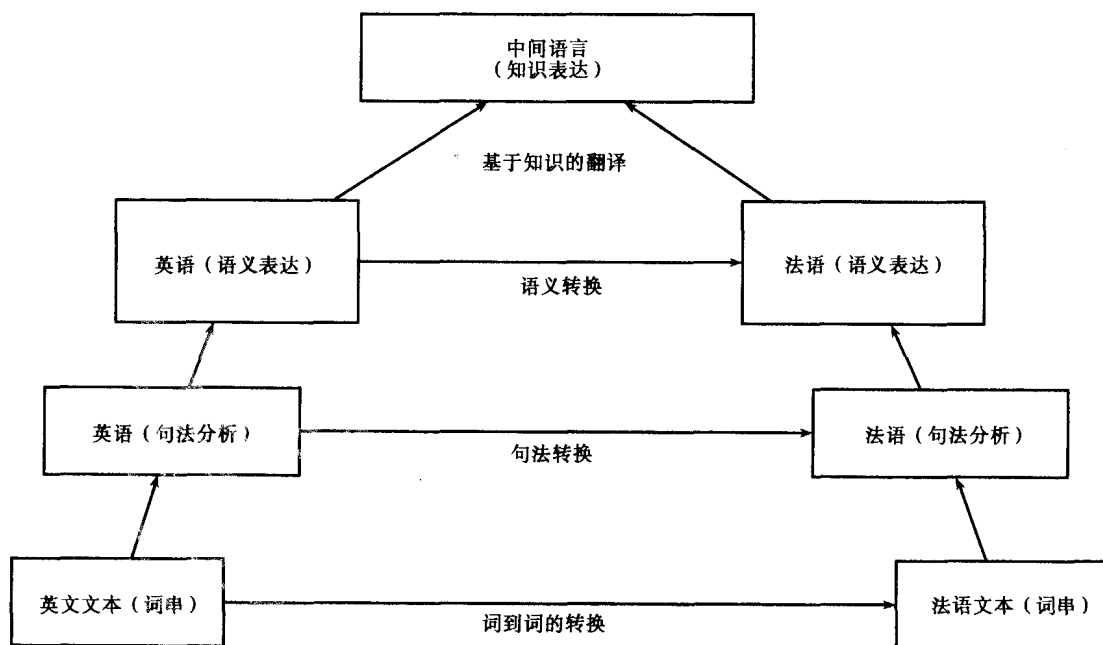


图 13.1 机器翻译中的不同方法(以英-法翻译为例)。基于词的翻译是将原文逐字翻译,转换翻译方法是首先将原文结构化,然后将其转换为目标语言的组织化结构,最后生成目标语言串。语义方法则是利用与句法树相比更为丰富的语义表示方法,比如量词范围消歧。最上端的中间语言法是通过一种独立于某种语言的知识表达进行翻译的方法(Knight 1997)

最简单的方法是直接翻译法,即“词”对“词”的对齐翻译方式。从源语言的表层句子出发,将词或固定词组直接置换成目标语言的对应成分,这种方法对翻译过程的认识过于简单化,明显的缺陷在于不同语言之间可能不存在一一对应的映射关系。词的歧义也是一个问题。例如,

在第 7 章中我们讨论了英文单词 *suit* 在法文中如何翻译,知道 *suit* 的确切译法需要依赖于它在文中取 *lawsuit* 还是 *set of garments* 的意思,需要参照具体的上下文来断定它的正确翻译方法。

直接翻译法的另外一个缺陷是语言中的词序(word order)问题。简单的词对词翻译方法显然会造成目标文本中词序的错误,句法转换(syntactic transfer approach)可以解决这个问题。首先对原文进行句法分析,然后用适当的手工定义规则将其转换成目标语言的句法树,最后在此句法树基础上生成目标语言的翻译。但是,在转换过程中,我们将不得不面临句法歧义的问题。

句法转换法虽然解决了词序问题,在一定程度上确保了翻译结果的句法准确性,但是句法的正确性并不等于语义的正确性。比如德文短句 *Ich esse gern* (*I like to eat*),从语法角度看它是“动词-副词”结构的,按字面意思直译为 *I eat readily* (或者 *willingly*, *with pleasure*, *gladly*),英语中没有类似的“动词-副词”结构能表达 *I like to eat* 的概念,所以句法转换方法并不能解决翻译中遇到的所有问题。

原文在经过句法分析之后,语义转换方法(semantic transfer approaches)将原文转化为语义表示形式,然后在此基础上生成译文,见图 13.11。语义转换方法能解决上述的句法结构不匹配问题,但基于语义转换的方法同样不是完美的翻译解决方案,即使字面上的意思翻译完全准确,但得到的最后译文对用户来说还是不易理解的。我们以英语和西班牙语的翻译为例(Talmy, 1985)对此进行说明,两者表达“方向”和“运动方式”概念的方式差异很大,在西班牙语中,“方向”概念是由具体的动词表达的,“运动方式”则通过另外的独立短语结构表达。

La botella entró a la cueva flotando. (13.1)

母语为英语的人也许能够勉强理解译文 *the bottle entered the cave floating* 的意思,但是如果译文中存在过多的类似句子,阅读起来无疑会很晦涩。我们知道英文中习惯用动词表示“运动方式”,用介词表示“方向”,所以正确的英文表达方式为:

The bottle floated into the cave. (13.2)

另外一种非直译的机器翻译方法是通过所谓的中间语言(interlingua)进行的。中间语言是独立于任何语言的知识表达形式,相对于前面的方法,中间语言方法的优点在于进行多语种翻译时,只需要对每种语言分别开发一个分析模块和一个生成模块,模块总数为 $O(n)$,而如果采用转换方法,就需要在每两种语言之间都开发一个转换模块,模块总数为 $O(n^2)$ 。但是,设计这样一种中间语言是一件难度很大的工作,起码要解决从一种语言转化为中间语言时所遇到的歧义问题,而且语言本身的完备性构建也是一件困难的事情。

本章的目标就是在上述机器翻译的过程中应用统计学的方法。从理论上说,图 13.1 中的每一步都可以用概率模型的方法解释,例如,“英语文本(词串)”到“英语(句法分析)”可以看成是概率句法分析的过程(参见第 11 章和第 12 章)。其他一些没有在图 13.1 中标识出来,但是对机器翻译是必不可少的步骤,例如语义消歧等,都能借助于统计方法。目前的绝大多数机器翻译系统采用的都是统计和规则方法相结合的策略,当然也有一些采用纯统计方法的翻译系统,我们将在下面的 13.3 节中介绍一个完全基于统计的翻译系统。

我们已经在前面的相关章节中详细地阐述了一些具体的方法,例如概率句法分析、语义消歧等。本章的重点是文本对齐(text alignment)问题,随后简单介绍词对齐(word alignment)的问题,最后说明一个统计翻译系统的构建过程。

13.1 文本对齐

目前,处理多语言文本的统计方法有很多,其中大多数都是针对文本对齐问题的,即相同的文字内容存在着不同的语言版本。一些国家,例如加拿大、瑞士、香港,可能有几种官方语言,那么其官方文件对应每种官方语言都有相应的版本,文本对齐就是建立这些不同版本之间的文本对应关系。虽然还有其他的多语言文本的资源,比如文学作品和宗教书籍,但是我们通常选取这类官方文件作为平行语料库(parallel corpus)。一方面是因为这类文本的数量很多,容易获得;另一方面,它们也满足了自然语言处理的准确性要求,因为这些官方文件都是遵照非常正式的写作风格撰写的,具备连贯一致性的特点。

在已知双语文本的情况下,首要任务是对齐文本,确定原文和译文句子或者段落之间的对应关系。一旦确定这种关系,下一步的任务是寻找词之间的对应,即建立一个双语词典,下面将逐一阐述这些具体问题。

13.1.1 句子和段落对齐

除了建立双语字典和进行机器翻译以外,文本对齐也是使用多语言语料库所必需的第一个步骤,而且对于其他利用多语言语料库的应用,例如语义消歧和多语言信息检索,这也是至关重要的一步。在现实生活中,很多产品说明手册都有不同语言的版本,随着产品的升级换代,说明书需要不断地更新。一个取巧的方法是,首先对齐新旧文档,找出其中的不同之处,然后对齐旧文档和其相应的翻译版本,只对改动部分进行翻译,这样就大大减少了译者的工作量。

在通常情况下,原文和译文中的句子是一一对应的,但是当人工翻译文档时,由于采用了意译法,甚至像技术类的文档,也有可能打乱原文的句子顺序。例如,图 13.2 中给出了两个英文句子及其法文翻译。虽然只是两个短句,仍然可以看出它们的语言组织结构迥然相异,而且有些部分甚至在译文中没有出现,例如英文 *achieved above-average growth rates* 在法语译文中没有出现,只有通过联系上下文,我们知道是在谈论“诸如可乐之类的软饮料销售额的增加”以后,才能隐含推导出这层意思。

简单地讲,句子对齐问题就是从句子内容出发,将源语言中的一组句子和目标语言中的一组句子进行对应的过程,每组句子可以为空,也可以额外加入对应源语言中不存在的句子,或者删除原有的句子,我们称这两组对应的句子为一个句珠(bead)。但是,什么样的句子是对齐的,即两个句子的重合需要达到什么程度?一般来说,如果两个句子之间只有个别词语对齐,那么不能说它们是对齐的,但是只要有任何子句(clause)对应出现,就可以判定句子之间的对齐关系。最常见的情况是源语言和目标语言中的单个句子之间的对应,称为“1:1”句子对齐或句珠。相关的研究表明 90% 的句子对齐都属于这类情况,有时候翻译者打乱句子的次序,会出现“1:2”或者“2:1”句子对齐,甚至“1:3”或者“3:1”句子对齐。

按照以上的框架定义,每个句子能且仅能出现在一个句珠中。图 13.2 中的第一个法语译句对应于英文原文的第一句,但是不能将其简单判断为“1:1”句子对齐,因为法语中的第二句翻译中的内容有很多也同样来自于第一句英文原文,准确地说它们应该是“2:2”句子对齐。我们知道,翻译时句子的次序可能被打乱,出现交叉依赖(cross dependency)的问题(Dan Melamed, p.c., 1998),下面介绍的算法都不能有效地解决这个问题。所以,如果文本对齐是定义在句

子层面上的,那么句子对齐只能是句子组之间的对齐,因此有必要区分对齐(alignment)和对应(correspondence)两个概念之间的差别。对齐不允许出现交叉情况,所以如果翻译时出现句子重新排序,只能将其说明为多对多的句子对齐,比如“2:2”,“2:3”或“3:2”。而且,翻译时为了理解的需要加入了一些句子或者出于疏忽的原因遗漏了一些句子,也可能出现“1:0”或“0:1”的对齐方式。

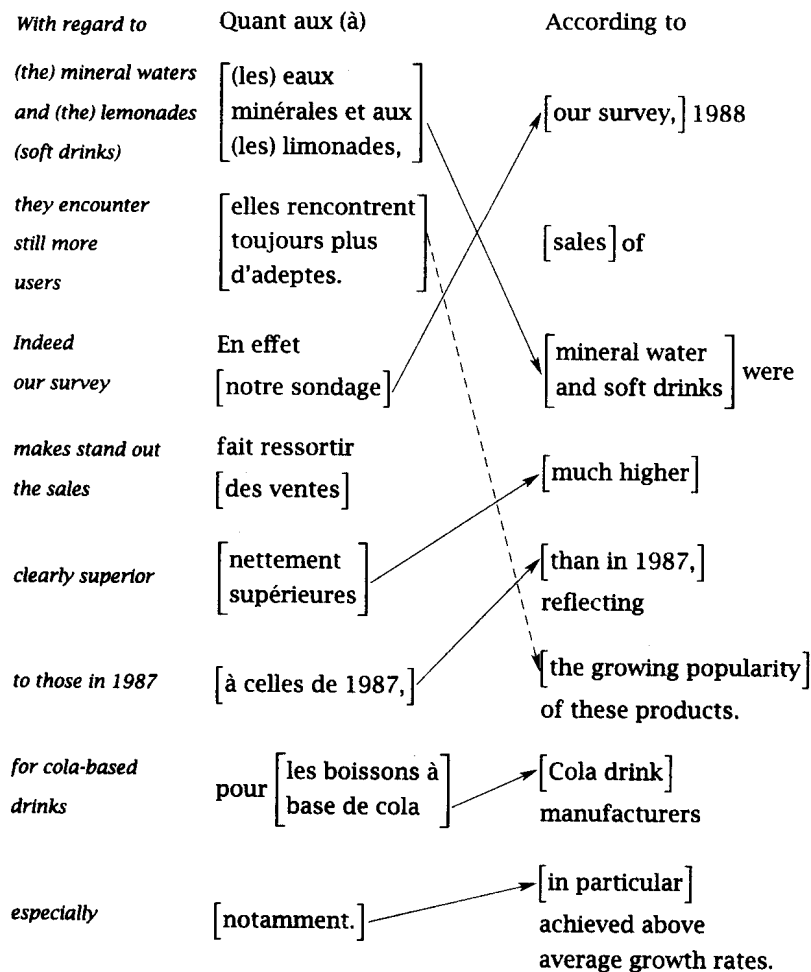


图 13.2 对齐文本。中间和右边两列分别是法文和英文句子,箭头标明了它们之间的对齐关系,左边斜体字部分是从法文直译得到的英文翻译

到本书落笔时为止,学者们提出了很多应用于平行语料库的对齐方式,如表 13.1 所示。从一方面来说,基本上可以分为基于长度的方法、基于词语对齐或字符串对齐的方法、基于偏移位置对齐的方法。另一方面,在对齐方式中可以分为位置之间的平均对齐和文本中某些特定位置的对齐。我们将分别介绍这些方法的特点,首先假设 S 代表源语言, T 代表目标语言,它们都是由单个句子组成的集合,即 $S = (s_1, s_2, \dots, s_I)$, $T = (t_1, t_2, \dots, t_J)$ 。如果有两种以上的语言需要处理,则可将它们转化为两两对应的问题。我们下面介绍的很多方法都利用了动态规划的方法,读者如果不了解其原理,建议首先阅读一下相关的书籍 (Cormen et al. 1990: ch. 16)。

表 13.1 句子对齐算法论文列表。表中列出了文本对齐采用不同方法的代表性论文,使用的语料库以及对齐所基于的信息类。

论文	语言	语料库	对齐所基于的基本信息类
Brown et al. (1991c)	English, French	Canadian Hansard	# of words
Gale and Church (1993)	English, French, German	Union Bank of Switzerland reports	# of characters
Wu (1994)	English, Cantonese	Hong Kong Hansard	# of characters
Church (1993)	various	various (incl. Hansard)	4-gram signals
Fung and McKeown (1994)	English, Cantonese	Hong Kong Hansard	lexical signals
Kay and Röscheisen (1993)	English, French, German	Scientific American	lexical (not probabilistic)
Chen (1993)	English, French	Canadian Hansard EEC proceedings	lexical
Haruno and Yamazaki (1996)	English, Japanese	newspaper, magazines	lexical (incl. dictionary)

13.1.2 基于长度的对齐算法

早期句子对齐的研究大都采用了比较平行语料中句子长度的方法,其基本原理是假设源语言和目标语言的句子长度存在比例关系,即原文中的短句对应于目标语言中的短句,长句对应于长句。虽然方法很简单,而且忽略其他很多文本中可利用的信息,但算法的效果相当好,而且效率很高。所谓的句子长度,我们将其定义为句子中单词或者字符的个数(针对英文而言),很多研究者都提出了各自的算法,下面将分别进行阐述。

Gale and Church(1993)

利用统计方法对齐句子就是使下式的概率值取最大值,假设对齐方式为 A , S 代表源语言, T 代表目标语言:

$$\arg \max_A P(A|S, T) = \arg \max_A P(A, S, T) \quad (13.3)$$

为了计算这个概率值,大多数方法是将对齐的文本分解为句珠序列 $(B_1, B_2, \dots, B_K)$,假设句珠的概率分布仅仅依赖于句珠内包含的句子,而各个句珠之间是独立分布的,我们可以得到下式:

$$P(A, S, T) \approx \prod_{k=1}^K P(B_k) \quad (13.4)$$

下面的问题就是在句珠内的句子已知的情况下,如何估算某一类句珠(例如“1:1”或者“2:2”的句子对齐)的概率值。

Gale and Church(1991;1993)采用的方法仅仅和句子的长度相关,句子的长度用其包含的字符数表示。如上所述,它假设源语言中和目标语言中的句子成比例,实验表明在采用直译方式并且语言之间的相似程度很大时,例如当两种语言属于一个语系时,这种方法相当有效。

Gale 和 Church 采用的语料库是 UBS(Union Bank of Switzerland)语料库,该语料中同时包含了采用英、法、德三国语言书写的相同内容的文档。文档中的段落结构很清晰,所以很容易实现段落对齐,个别出现混淆的地方可以通过人工方法进行校对。段落对齐在这个方法中很重

要, Gale and Church(1993)的实验表明,在非段落对齐文本中的错误要比段落对齐文本中的大三倍左右,如果不采取人工校对法,他们建议将这个算法分两步完成,第一步利用算法对齐文本中的段落,第二步对齐段落中的句子。Shemtov(1993)利用动态规划的思想实现了这个算法,并且它对在翻译中可能出现的段落层次上的插入和删除情况处理得也很好。

Gale and Church(1993)利用句子的长度来评估两组句子之间的对齐程度,他们提出的句子对齐算法中假设只存在五类句珠模式,即 $\{1:1, 1:0, 0:1, 2:1, 1:2, 2:2\}$ 。这个假设前提使得利用动态规划方法寻找双语文本之间的“最短距离”很方便,换言之,如果假设 $D(i, j)$ 表示两组句子 $s_1, s_2, \dots, s_i$ 和 $t_1, t_2, \dots, t_j$ 之间对齐的最小耗费函数(cost function),那么我们可以利用递归法反复计算这个值,并假定 $D(0, 0) = 0$, 下面为计算公式:

$$D(i, j) = \min \begin{cases} D(i, j-1) + \text{cost}(0:1 \text{ align } \emptyset, t_j) \\ D(i-1, j) + \text{cost}(1:0 \text{ align } s_i, \emptyset) \\ D(i-1, j-1) + \text{cost}(1:1 \text{ align } s_i, t_j) \\ D(i-1, j-2) + \text{cost}(1:2 \text{ align } s_i, t_{j-1}, t_j) \\ D(i-2, j-1) + \text{cost}(2:1 \text{ align } s_{i-1}, s_i, t_j) \\ D(i-2, j-2) + \text{cost}(2:2 \text{ align } s_{i-1}, s_i, t_{j-1}, t_j) \end{cases}$$

图 13.3 中给出了一个计算对齐耗费的例子,动态规划方法计算了所有可能的情况,并且找出其中具备最小耗费的 $D(I, J)$ 。因为动态规划算法是二次的,并且将段落作为“锚”,所以它的运算效率很高。

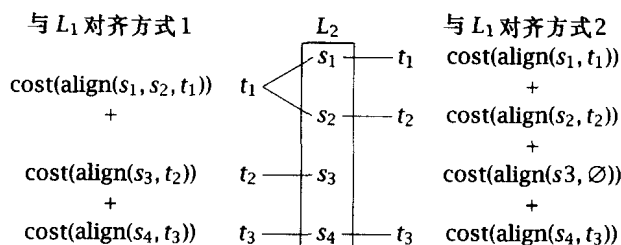


图 13.3 计算对齐的耗费,图中给出了两种不同对齐方式的计算结果

下面一个步骤就是确定每类对齐的耗费。耗费的计算基于句珠中每一种语言句子的长度 l_1 和 l_2 , 即句子中的字符数,同时把一种语言中的每个字符相对于另一种语言对应出现的字符数作为随机变量,假设这些随机变量服从正态分布,并且满足独立同分布。我们从实际语料中估计这些参数值,以均值 μ 为例,它的值大概为 1,因为实验表明平行语料中对应文本长度的比值和 1 接近,德文/英文 = 1.1, 法文/英文 = 1.06, 其方差 s^2 用长度差值的平方近似。

如上所述的耗费由两种语言文本句子组之间的距离度量决定。设句珠中源语言和目标语言的句子长度分别为 l_1 和 l_2 , 它们之间的距离度量就定义为 δ , δ 的计算考虑到了句子组长度差异和全部语料库均值和方差:

$$\delta = (l_2 - l_1 \mu) / \sqrt{l_1 s^2}$$

其耗费计算函数如下:

$$\text{cost}(l_1, l_2) = -\log P(\alpha \text{ align} | \delta(l_1, l_2, \mu, s^2))$$

其中参数 $\alpha \text{ align}$ 代表某种可能的对齐模式,例如(“1:1”, “2:2”等)。取负值是为了表达距离概念,使得出的概率值和距离成反比。上式可以通过贝叶斯公式计算,即 $P(\alpha \text{ align}) P(\delta | \alpha$

$align$), $P(\alpha align)$ 相当于对齐模式的先验概率, 例如我们可以给“1:1”模式较高的概率值, 因为大多数情况下都是一对一的对齐模式, 具体的值可以通过统计得到。

所以, 从本质上说, 这个算法尽可能地在长度相近的句子之间建立了对齐关系。实验表明, 它在 UBS 语料库上效果很好, 一般错误率在 4% 左右, 改进后其错误率甚至可以达到 0.7%。这种算法在确定“1:1”对齐模型上性能最好, 但是对那些比较难匹配的模型, 例如在“1:0”和“0:1”模式上的对齐结果, 基本上都是错误的。

Brown et al. (1991c) 的方法

Brown et al. (1991) 提出的方法和 (Gale and Church, 1993) 的很相似, 但以句子包含的单词数而不是字符数来衡量句子长度。按照 Gale 和 Church 的观点, 这种句子长度的估算方法并不可靠, 因为相比较而言, 翻译为目标语言之后句子中单词的数量可能变化很大, 但是我们应当注意到他们的研究目标的不一致性。Brown 等人的研究目的并不在于对齐全部文本, 而仅仅是希望得到一个对齐的语料子集, 以备进一步的研究之用。在段落对齐上, 该算法利用了词锚 (lexical anchor) 概念, 即利用一些有固定译法的单词或者短语结构, 删除在语料中没有充分对齐的段落。该算法采用加拿大议会的双语文献语料库, Brown 发现在该平行语料库中经常会出现段落的错位现象, 在处理中他忽略了这种类型的段落。当然, 除了如何计算句长之外, 该算法和基于长度的对齐算法还存在其他的差异, 比如在 Brown 的算法中估计模型参数利用了 EM 算法 (见 13.3 节), 实验结果表明其性能很好, 尤其是在“1:1”对齐模式上。要注意的是, 会出现一些短段落对齐错位现象, 主要原因在于算法只考虑了句长而忽略了词的识别。

Wu (1994) 的方法

Wu (1994) 将 Gale and Church (1993) 的对齐应用到英文和繁体汉字 (香港) 文本对齐问题上, 研究发现, 当应用到两类不同语系的语言时, 算法的效果并不是很令人满意。但是, 除了一些标题段落, Wu 的报告结果和 Gale 和 Church 的结果相差并不太大。为了提高准确率, Wu 利用了一些词的信息, 具体的方法将在 13.1.4 节进行详细讨论。但是我们注意到, Wu 的测试文档中的句珠模式很多是“3:1”, “1:3”和“3:3”。对于这类模式, 其实很多算法都不能很好地将它们对齐, 当然也包括 Wu 的算法。

13.1.3 基于信号处理技术的偏移位置对齐算法

和其他方法不同, 基于信号处理方法的偏移位置对齐算法没有试图对齐句子, 而是在平行文本中利用位置偏移量的概念, 即源文本中一定位置的文本和目标语言中一定位置的文本是大致对齐的, 我们给出两种采用此思路的算法。

Church (1993) 的方法

Church (1993) 认为我们上面介绍的各种基于长度的对齐算法只适合处理“干净”的文本, 例如加拿大议会的文档, 它们都遵循一定的写作规范。但是, 如果将其用于“脏”文本的处理, 例如 OCR 识别输出的文件, 算法的出错概率会很大, 因为 OCR 识别输出文本经常遗漏段落分割标记、标点符号以及诸如脚注、表等文本信息, 而这些信息对界定段落和句子边界至关重要; 电子文档虽然避免了这些缺陷, 但是内部经常包含未知结构标记等“噪声”, 影响对齐效果。

Church 的算法利用了同源词(cognate)信息。所谓同源词,就是不同语言中的词由于借鉴或者来源于同一种祖先语言而具有的相同特征。例如法语单词 *supérieur* 和英文单词 *superior*,但是 Church 并没有按照一般方法寻找源语言和目标语言中的同源词信息(Simard et al. 1992)或者寻找对应的词,而是通过字符序列的相似程度寻找同源词,显然效果的好坏要取决于源语言和目标语言中相同字符序列的个数。Church 认为,只要某种语言文本中存在很多名称词和数字词,该算法不仅适用于那些发源自罗马字符的语言,而且对其他语言也同样有效。

Church 首先将源语言和目标语言连接起来,构建一个点阵图(dot-plot),即如图 13.4 所示的正方图。图中两个轴都代表连接后的文本,坐标 (x, y) 处的点表示连接文本中位置 x 与位置 y 处的文本匹配,Church(1993)将 4-gram 作为匹配的最小单位。各种现有的信号处理方法都可以用来进行点阵图的压缩。上述点阵图的一个特点就是图中的对角线(图中已经做了说明)由文本中的匹配点组成,我们称之为双语文本映射(bitext map),其中左下方和右上方的两条对角线文本对齐尤为重要,代表了源语言和目标语言文本之间的匹配关系,它们之所以出现是由于文本之间存在同源词的缘故。这里不探讨算法的细节,实际上利用一些技巧可以使我们避免计算整个点阵图,另外再对不同的 n -gram 建立倒排权重,即权重与其出现频度成反比。实验表明它的错误率很小。由于这种算法并没有句珠概念,所以我们无法将它的结果和以上基于长度的算法进行量化比较。此外,这种算法并不是一种通用算法,有一定的局限性,因为它的假设前提是源语言和目标语言中存在很多相同的字符序列,所以当应用到不同语系的机器语言翻译时,例如西欧语系和远东语系时,效果可能并不是我们所期望的,因此在原文和译文之间可能存在很少甚至完全没有相同的字符串。

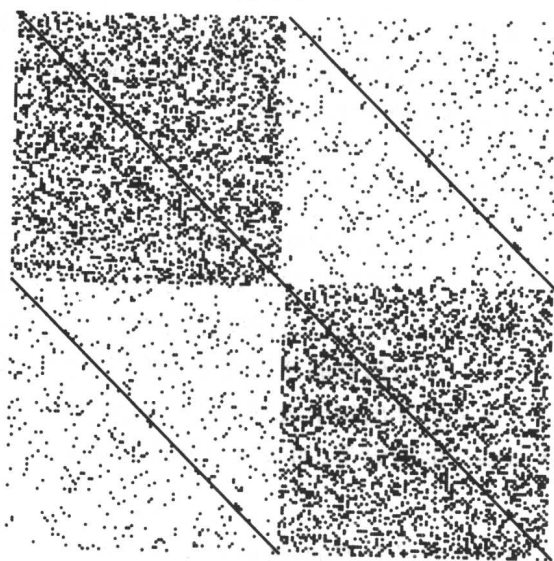


图 13.4 点阵图的一个样本。源语言和目标语言文本被连接在一起,如果位置 x 和 y 处的文本存在对应关系,则在图中用一个点来标记坐标 (x, y) 。相比较而言,显然语言与其本身的相关程度会比较大,所以图中左上方和右下方的颜色比较深,图中左上方和右下方的对角线表明了自身的对应关系,左下方和右上方表明了源语言和目标语言之间的对应关系

Fung and McKeown(1994)的方法

在 Fung and Church(1994)早期工作的基础上,Fung and McKeown(1994)希望能够设计出适用于以下三种情况的算法:(i) 无需考虑句子的边界;(ii) 双语文本只是局部对齐,即源语言和目标语言文本之间存在很多没有对齐的段落;(iii) 与具体的语言无关,比如,Fung 和 McKeown 采用的是英文和繁体中文(香港)的语料库。这里首先需要导出一个小规模的双语词典,词典提供了一些对齐的基点。具体构建方法如下:每个词对应一个信号,用到达向量(arrival vector)表示,标记了词的不同出现位置之间的词数^①。例如,假设一个词出现 4 次,位置向量为(1, 263, 267, 519),词的到达向量可以表示成(262, 4, 252)。对英文和中文词都做同样的处理。可以设想,如果两个不同语言的词之间的出现频率和位置差别都很大,那么它们之间对齐的概率很小,否则就可以使用动态规划算法计算它们的相似度。算法中采用的是一种标准的动态规划方法,称为动态时间伸缩(Dynamic Time Warping),这是解决通过语音识别不同长度信号的对齐问题所采用的常用算法(Rabiner and Juang 1993:sec.4.7)。在所有的英/中文词对中,可以选取相似度大的词对组成双语词典。连接英文和繁体中文文本,采用与 Church 类似的点阵图并标记对齐的词对。下一个步骤就是再次利用动态规划方法寻找源语言和目标语言文本的对齐。但是,我们注意到这个步骤事实上和具体的语言无关。另外,由于算法过程中利用了词的信息,所以对语言中的词相当敏感。

13.1.4 句子对齐的词汇方法

我们知道,前面基于长度的对齐方法的一个显著缺陷就是不能有效地处理噪声文本,所以有些研究者提出了上面的基于文本偏移位置对齐的方法,以避免这个问题。本节中我们在算法里融入词信息,提出健壮性更好的句珠对齐算法。

Kay and Röscheisen(1993)的方法

Brown et al. (1991c)和 Gale and Church(1993)早期的方法中很少或者根本没有用到句子中的词信息。但是,显而易见,句子中的词能给句子对齐提供很多有价值的信息,特别是当句子的长度近似时(这种情况在列表中尤为常见),这种信息非常重要。Kay 和 Röscheisen 于是利用部分词语对齐来得到句子对齐。词语信息的使用意味着不需要更高级别的段落对齐。这种方法是使用词语级别的部分对齐来推导出句子级别的对齐的最大似然估计方法,同时又反过来优化词语对齐。词语对齐所基于的假设是,如果两个词语的分布相同,那么它们是对应的。算法的具体步骤如下:

- 假设双语文本的首句和尾句是对齐的,将它们作为初始锚。
- 重复以下过程直至绝大多数句子对齐:

1. 对源语言和目标语言的句子列表做类似“点积”的运算,其结果为句对集合,我们形象地称之为“信封”(envelope),句对中各有源语言和目标语言的一个句子。假设“信封”中的句对是对齐的。如果它们跨越某个锚匹配或者各自和锚的距离差

^① 汉语文本中词的边界没有明显的标记,需要进行分词处理。

别过大, 则从“信封”中删除此句对。其中, 距离判断条件可以随着句子和锚之间的距离的扩大而有所放宽, 图 13.5 中给出了图例。

2. 选取句对同现的词对, 词对中的两个词分布模型近似, 即如果一个词出现在绝大多数句子中, 另外一个词也要在其对应的句子中出现。
3. 寻找包含很多同现词对的源语言和目标语言的句对。如果句对中包含很多这样的词对, 将其中可信度最高的词对中的词加入锚集合, 并转向(1)。

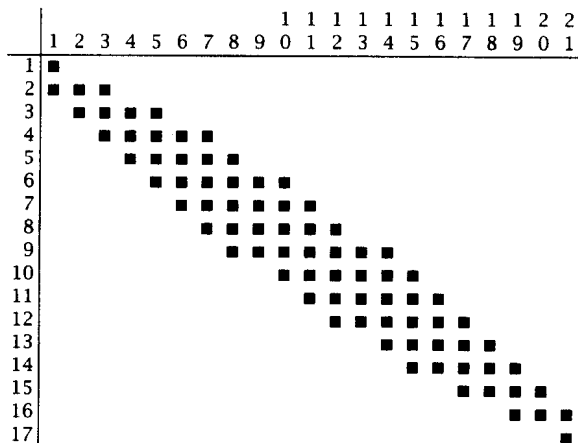


图 13.5 枕状“信封”(pillow-shaped)图示, 源语言和目标语言的句子集合用 L_1 和 L_2 表示, 分别对应于图中的纵坐标(1-17)和横坐标(1-21), 并假定 L_1 和 L_2 的开始以及句子(17,21)为锚, 图中“■”表示在当前的算法迭代中对齐的句对(Kay and Röscheisen 1993: figure 3)

算法的准确度取决于“退火程序”过程, 如果每次迭代我们都降低词对可信度判断的条件, 那么每次加入的锚数目将增加, 算法需要的迭代次数可能很少。根据一般的观察, 经过五次迭代就能得到满意的结果。注意, 这个算法具备健壮性的特点, 因为它并没有限制对齐的模式, 例如“脏”句子可以没有任何句子与其对齐。Kay and Röscheisen(1993)以 American articles 文档作为语料, 结果显示经过四次迭代后 96% 的句子会大致对齐, 其他没有对齐的模式主要是“1:0”和“0:1”。1000 句的议会文档经过五次迭代后对齐错误的句子大概有 7 个, 其中 5 个还是由于句子边界界定算法的错误间接引起的。另一方面, 应当指出的是, 这种算法的计算量很大, 并且如果某些大的段落在翻译过程中缺失或者对它的次序进行了重排, 应用枕状“信封”可能会限制搜索的范围, 从而遗漏某些正确的对齐句子。

Chen(1993)的方法

Chen(1993)提出了一种基于词-词翻译模型的句子对齐算法, 假定翻译模型已知, 最佳对齐模式就是对生成语料的极大似然估计。算法过程同样借用了动态规划的算法, Chen 认为基于长度的句子对齐算法健壮性不足, 而以前的基于词信息的方法又显得效率太低。通过简单的翻译模型和设定阈值的方法, Chen 的算法在一定程度上克服了以上的缺点。

除了利用翻译模型估计对齐的耗费函数以外, Chen 的算法模型非常类似于 Gale and Church(1993)提出的方法。所以, 为了对齐源语言文本 S 和目标语言文本 T , 我们首先将它们分解为句

珠集合 B_k , 每个句珠包含零个或者多个句子, 所以实际上句珠序列覆盖了整个语料库的文本。

$$B_k = (s_{a_k}, \dots, s_{b_k}; t_{c_k}, \dots, t_{d_k})$$

假设句珠之间相互独立, 则最优的句珠序列 $A = B_1, \dots, B_{m_A}$ 满足下式:

$$\arg \max_A P(S, T, A) = \arg \max_A P(m_A) \prod_{k=1}^{m_A} P(B_k)$$

其中 $P(m_A)$ 表示生成 L 个句珠的概率, 假设其概率分布为均匀分布, 如果 L 的值超过语料库中句子的数目, $P(m_A)$ 的概率都为零。

这样, 算法就转化为如何寻找合适的翻译模型的问题, Chen 认为出于简单性和算法效率的考虑, 应该尽量选择简单的模型, 因此算法中忽略了词序信息以及源语言中的词可能对应目标语言中多个词的情况。算法利用了词珠概念(word bead), 并将词珠模式仅限于“1:0”, “0:1”和“1:1”三种类型。翻译模型的基本思想是, 如果一个词多次被翻译为目标语言中的同一个词, 则相应为“1:1”词珠对齐模式的概率值就比较高, 远大于其“1:0”和“0:1”词珠模式的概率值。我们这里不讨论翻译模型的具体形式, 因为它和 13.3 节中的翻译模型类似。计算对齐概率时, 算法并不需要计算所有词珠的组合形式, 而只选取其中的最优匹配方案。实际上, 它是通过贪婪算法得到的。算法首先假设“1:0”和“0:1”对齐模式, 然后不断地用“1:1”的对齐模式代替它, 使对齐的概率改善最大, 直至没有任何改善效果。

Chen 通过 EM 算法训练模型参数^①, 首先需要人工校对对齐 100 个句子, 然后采用渐进式的 EM 算法在每种语言的未标注语料上(20 000 个句子)重新估计模型参数。算法的最后一步利用动态规划方法确定最优的对齐模式。但是, 为了加快算法的运算速度, 争取算法在线性时间内完成, 它采用了分支定界搜索法, 只存储局部前缀的对齐结果, 但是当翻译过程中插入和删除的文本过大时, 算法会出现错误。vannilla 动态规划算法可能更适用于这类情况, 读者可以参阅 Simard and Plamondon(1996)。Chen 的算法设计了一个特别的过程处理这种情况, 它在遇到类似删除情况时起到重新定位的作用, 使正常的对齐程序开始于正确的位置。

这种算法已经被应用到这样的大规模文本的对齐问题: 包含英语和法语两种版本的加拿大议会文档以及欧洲经济社团期刊的百万量级句子的语料库。Chen 的算法结果表明, 该算法在整个文档集合上的错误率大概为 0.4%, 好于其他的算法, 而且这些算法都是基于这个语料库的子集的。Chen 认为大多数对齐错误是由于使用了性能不高的句子边界界定算法而导致的, 所以通过改进翻译模型以提高对齐正确率不太可能, 相反只会降低算法的运行速度。另外我们注意到, 现有的算法都预定义了对齐模式, 即只限于“1:0”, “0:1”, “2:1”或“1:2”等几种模式, 所以在处理平行文本中出现的其他对齐模式时, 算法不可避免地会出现错误。虽然 Gale 和 Church 没有提出解决这个问题的更好方法, 但是扩展当前的翻译模型, 使其适用于多种对齐模式, 并不困难, 而 Chen 的论文中也没有指出扩展模型后是否会增加错误率。

Haruno and Yamazaki(1996)的方法

Haruno and Yamazaki(1996)认为当源语言和目标语言结构迥异时, 以上所有算法在对齐短

① 在标准的 EM 算法中, 对于每一个数据, 综合了所有路径的期望值作为参数, 但是出于计算量的考虑, 我们只选择最优路径, 我们称之为采用 Viterbi 方法的 EM 算法, 这是一种启发式算法, 效率很高。

文本时性能都不高。他们提出的对齐算法实际上是 Kay and Röscheisen(1993)算法的变种。但是,论文中阐述了一些令人感兴趣的观点^①,首先他们认为语法结构相差很大的语言,比如日文和英文,在词匹配时功能词会影响算法的匹配结果,所以在算法中只考虑实词之间的匹配。算法用词性标注程序对语言中的单词进行分类,剔除语言中的虚词。然后,利用在线词典寻找匹配的词对,在短文本中不能保证足够的数量,因而不能使用 Kay 提出的方法。与以前的算法不同,在一定程度上它们都可以看做基于知识的方法。算法从实用角度出发,利用了诸如词性分类器、在线词典等资源,如果置这些丰富的资源于不顾,显然有些浪费。但是另一方面,正如 Haruno 和 Yamazaki 所指出的,字典并不能完全代替文本中的词对齐的作用,特别是在技术类文档中,因此应该有效地结合两种方法,即使源语言和目标语言之间差异很大,处理短文本时也能取得很好的效果。

13.1.5 小结

上面的所有算法表明,如果我们处理的是“干净”的文本,句子对齐并不是一个很难解决的问题,相当多的算法都可以取得很好的结果。但是,当翻译通常都采用意译法,而且源语言和目标语言为不同的语言体系时,基于词模型的方法显示了其优越性,算法有很好的健壮性和通用性。在算法中应用信号处理或者整句对齐的方法只能实现源语言和目标语言的近似匹配。但是,面对具体问题,选取什么算法还应该综合考虑各方面的因素,例如对对齐准确率的要求、实际应用和处理何种语言等。

13.2 词对齐

对齐文本的典型用途是建立双语词典和术语数据库,通常分两个步骤完成。第一个步骤是在文本对齐的基础上实现词对齐(在有些算法中文本对齐和词对齐的过程是结合在一起的)。一般的方法是利用词对出现的频率,频率越高,被收入双语词典的可能性越大。例如, *adeptes-products* (见图 13.2)对齐的情况在语料中只出现了一次,不应该被收入双语词典中,因为联系上下文 *adeptes* 的正确翻译应为 *user* 而不是 *products*。

我们在 5.3.3 节简短地给出了一种利用关联度量的词对齐方法,例如 Church and Gale (1991b)使用的度量 χ^2 ,当阈值很高时该方法还是很有效的。但是,在某些情况下这些关联度量也容易产生误导,特别是当 L_2 中有多于一个的目标词与 L_1 中的原词对应时。例如,单词 *house* 可能被错误地翻译为 *communes*,而不是 *chambre*,主要原因就在于包含 *house* 的句子在被翻译为法语时,和 *Chambre de Communes* 同现的次数很多。

但是,如果我们在关联度量信息之外加入一些额外的信息,例如假设词对之间的翻译满足一一对应关系,虽然这个假设只对一部分词来说是正确的,但是研究表明结果的准确率还是相当高的(Melamed 1997b)。绝大多数基于这个假设的方法都采用了类似 EM 的算法,例如前面介绍的 Chen(1993),Dagan et al.(1993),Kupiec(1993a),Vogel et al.(1996)以及下面将要介绍的

① 另一方面,算法中的一些细节存在问题。例如,利用互信息评估词之间的匹配程度(5.4 节对互信息有专门的讨论,指出当词频较低时,通过 t 统计法过滤不可靠的互信息仅仅是一种辅助方法),利用专门计分函数把字典和语料的统计信息结合起来。

Brown et al. (1990) 的算法。算法内部包含了几个迭代计算的过程,通过已知对齐词序列重新计算加入的对齐词,再在此基础上重新计算对齐词序列。另外,一些研究者为了建立术语数据库,还致力于短语的对齐,它要比词对齐复杂得多(Wu, 1995; Gaussier 1998; Hull 1998)。总的来说,这些迭代性质的算法的效率要比单纯的关联度量方法差一些。

我们可以毫不夸张地说,今后词对齐算法应该更多地利用先验知识,比如双语语料库,而不是试图从对齐语料本身入手搜寻可利用的信息,Klavans and Tzoukermann (1995) 提出了类似的算法,有兴趣的读者可以阅读该论文。

13.3 统计机器翻译

在本书的 2.2.4 节中,我们介绍了噪声信道模型的概念,它在自然语言处理中的一个重要应用领域就是机器翻译,如图 13.6 所示。为了将法文翻译为英文,我们建立了一个噪声信道模型,并以英文句子 e 作为输入,翻译模型将它转化为法文句子 f ,利用解码器寻找最有可能与其对应的英文句子 $\hat{e}$ 。

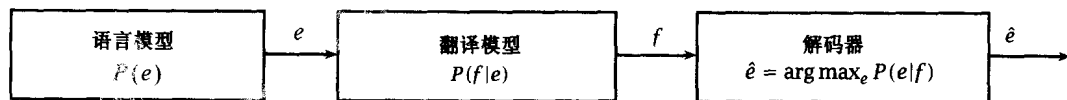


图 13.6 机器翻译的噪声信道模型。语言模型生成英文句子 e , 翻译模型将其转化为法文句子 f , 解码器的作用在于确定最有可能和 f 对应的英文句子 $\hat{e}$

这样,在从英文到法文再到英文的整个翻译过程中,一共分为三个部分:语言模型、翻译模型和解码器。但是模型的参数(即翻译概率)是未知的,需要我们估计它们的值。

语言模型(Language model):语言模型实际上给出了英文句子 $P(e)$ 的生成概率,在第 6 章、第 11 章和第 12 章中详细阐述了如何基于 n 元语法以及概率句法分析建立语言模型,所以这里假定事先已经有一个定义良好的语言模型。

翻译模型(Translation model):下式给出了一个简单的基于词对齐的翻译模型:

$$P(f|e) = \frac{1}{Z} \sum_{a_1=0}^l \cdots \sum_{a_m=0}^l \prod_{j=1}^m P(f_j|e_{a_j}) \quad (13.5)$$

我们借用了 Brown et al. (1993) 中的符号表示方法: e 代表英文句子, l 是按词计算的句子长度; f 是法文句子, m 是其长度; f_j 是 f 的第 j 个词, a_j 是 e 中与 f_j 对齐的词的位置; e_{a_j} 是 e 中与 f_j 对齐的词; $P(w_f|w_e)$ 是翻译概率(translation probability),即在已知 w_e 在英文句子中的条件下, w_f 出现在对应法文句子中的概率, Z 是归一化常数。

上述公式所包含的基本思想很直观,它累加了法语单词对齐英语单词所有情况的概率, $a_j = 0$ 表示法语句子中的第 j 个单词在英文中没有对齐成分(empty cept)。此外,我们注意到每个英文单词可以对应于多个法语单词,但每个法语单词至多仅和一个英文单词对应。

假设各个单词的翻译之间是无关的。句子对齐以后,对所有 m 个对齐概率求乘积作为对齐模式的概率。例如,如果我们想计算下面两句的对齐概率,即:

$$P(\text{Jean aime Marie} | \text{John loves Mary})$$

其中 $(Jean, John), (aime, loves), (Marie, Mary)$ 代表对齐成分, 那么只需要将三个单独的概率值相乘:

$$P(Jean|John) \times P(aime|loves) \times P(Marie|Mary)$$

总之, 我们通过累加所有对齐概率来计算 $P(f|e)$ 的值。在每种对齐方案中都假设了两个基本前提: 每个法语单词仅和一个英语单词(或没有英语单词)对应; 句子中每个法语单词的生成是相互独立的^①。

解码器 (Decoder): 我们在 2.2.4 节中已经介绍了几个解码器的例子, 这里也采用类似最大化的方法, 并且在公式中忽略了 $P(f)$ 的值, 因为我们注意到它的值是固定的。

$$\hat{e} = \arg \max_e P(e|f) = \arg \max_e \frac{P(e)P(f|e)}{P(f)} = \arg \max_e P(e)P(f|e) \quad (13.6)$$

问题的关键在于它的搜索空间是无限的, 所以我们必须采用启发式的搜索算法。栈搜索法是一个可取的方法(参见 12.1.10 节), 其基本思路是渐进地构建英文句子, 算法过程中随时保持一个局部的翻译, 每次在这个局部翻译的基础上增加少数单词和对齐模式。如果当前扩展正确的可能性很小, 可以通过剪裁栈返回以前的状态。虽然算法运行效率很高, 但是它的一个缺点是不能保证找到全局的最优翻译。

翻译概率 (Translation probabilities): 我们用 EM 算法估计翻译模型的参数(EM 算法可参阅 14.2.2 节), 假设平行语料库中的句子已经对齐。

在前面的章节中讨论了词对齐的问题。估算词之间是否对应的一种方法是计算关联度量值, 例如统计量 χ^2 。但是, 如果不考虑源语言和目标语言中词的一对多情况, 可能产生很多“假”对应的词对, 例如前面提到的 $chambre \leftrightarrow house, chambre \leftrightarrow chamber$ 。

EM 算法的基本思路是它解决信任度分配(credit assignment)问题。如果某个词和目标语言中的某个特定词对齐的可能性很大, 则完全可以绑定它们, 从而避免“1:2”或“1:3”的对齐模式以及过多的没有对齐的词。

我们首先随机设定 $P(w_f|w_e)$ 的初始值, 假设英文单词 w_e 已知, 在 EM 算法的 E 步骤中计算 w_f 在法语句子中出现次数的期望值:

$$Z_{w_f, w_e} = \sum_{(e, f) \text{ s.t. } w_e \in e, w_f \in f} P(w_f|w_e)$$

其中求和公式遍历所有含有 w_e 和 w_f 的英语和法语句子, 为简单起见, 我们假设一个词在句子中仅出现一次。

EM 算法的 M 步骤在期望值的基础上重新估计翻译模型参数:

$$P(w_f|w_e) = \frac{Z_{w_f, w_e}}{\sum_v Z_{v, w_e}}$$

其中求和公式遍历了所有的法文单词 v 。

这仅仅是 Brown et al. (1990) 和 Brown et al. (1993) 中的算法的简化版本, (Kupiec, 1993a) 给出了利用 EM 算法实现对齐的详细过程。在上面的介绍中我们省略了一些必要的过程, 比如

① 从另一个角度来看, 这意味着一个英语单词可以对应多个法语单词。

可能性比较小的对齐显然应该给予一定的惩罚。例如,如果位于句首的英语单词和对应的法语句子中的最后一个词对应,那么这种位置对齐的“扭曲度”(distortion)应该在一定程度上降低了它们之间对齐的概率。

与此类似,我们引入了“富余度”(fertility)的概念。“富余度”表示翻译时和一个英文单词对应的法语单词的数目。在没有受限的模型中,我们无法区分是否每个法语单词由不同的英文单词生成,或者所有的法语单词由同一个英文单词产生。而“富余度”的概念则有助于辨别源语言和目标语言文本中的对齐信息,例如语料中的英文单词 *farmers* 对应于法语的 *les agriculteurs*,所以它的“富余度”等于 2。当然,绝大多数英文单词的“富余度”为 1,即一个英文单词翻译为法语后同样为一个法语单词。

在已对齐的英国议会语料库上评估模型,会发现只有 48% 左右的法语句子解码(或者称为翻译)正确。错误主要分为两类,一类是解码器导致的错误(13.7),另一类是语法结构解码错误(13.8),参见(Brown et al. 1990:84)。

- | | | |
|------------|---|--------|
| a. 源句 | Permettez que je donne un exemple à la chambre. | |
| b. 正确翻译 | Let me give the House one example. | (13.7) |
| c. 不正确解码 | Let me give an example in the House. | |
| a. 源句 | Vous avez besoin de toute l'aide disponible. | |
| b. 正确翻译 | You need all the help you can get. | |
| c. 不合语法的解码 | You need of the whole benefits available. | (13.8) |

详细的讨论可以参见 Brown et al. (1990, 1993), 其中提出了翻译模型所面临的几个问题:

- “富余度”的不对称性。以英语和法语的翻译为例。单个法语单词可以同时对应数个英语单词,例如英语单词 *to go* 翻译为法语时对应 *aller*, 目前所有的对齐算法都不能处理此类情况。当然模型可以建立模块对其进行单独处理,即将单词 *go* 对应 *aller*, 而 *to* 则不对应翻译中的任何一个语言单词。但是,这种做法充其量是基于具体实例的,不能用来解决一般性的问题。

注意,一个单独的英文单词可以对应多个法语单词。例如,上面提到的英文单词 *farmers* 对应于法语的两个词 *les* 和 *agriculteurs*, 所以 *farmers* 的“富余度”为 2。这样就会出现“富余度”的不对称性。

- 独立性假设。和其他很多统计自然语言处理的应用一样,我们在构建概率模型时做了很多独立性假设,而这些假设并不是严格满足的。在翻译模型中,由于取概率乘积的原因,显然模型会倾向于选择长度短的句子。我们可以在概率计算公式中引入常数 c' , 它的大小随着句子长度 L 而增加。当然,一个更好的模型不应当基于这种独立性假设,有兴趣的读者可以参阅 Brown et al. (1993:293), 在 12.1.11 节中也有相应的阐述。
- 对训练数据的敏感性。模型或者训练数据的细微变化,比如训练数据取自于语料的不同部分,会导致最后参数估计值的较大变化。例如, Brown et al. (1990) 中的模型训练的 $P(\text{le} | \text{the})$ 值为 0.610, 而 Brown et al. (1993:286) 中的模型为 0.497, 虽然我们不能就此得出这些微小差异会给最后的结果带来很大影响的结论,但是为了获得满意的结果,我们不得不考虑训练数据和具体应用之间需要什么样的内在关系的问题。读者可以参阅

10.3.2 节中关于词性标注应用的讨论,其中对训练数据和应用语料之间的差异所引起的结果变化进行了探讨。

- **算法效率。**因为算法处理长句会花费过多的时间,所以包含 30 个词以上的句子要从训练数据中删除,可参阅(Brown et al. 1993:282)。

从表面上看,以上这些问题似乎都和模型相关,但是它们都可以归于模型中语言学知识的缺乏。例如,如果在翻译模型中引入句法分析,就可能估计出子句中各部分相互之间的相关性,而不是简单地通过“富余度”的方法来推算。一个好的模型应该避免过多的独立性假设,应该更好地利用训练数据以及减少需要搜索的空间大小。

Brown et al. (1990) 和 Brown et al. (1993) 发现了另外一些问题并直接指明了系统由于缺少语言学知识而导致的翻译错误。

- **没有引入短语概念。**模型本身只和单个词语有关,但是如果能处理短语之间的关系,例如上面提到的 *to go* 和 *aller*, *farmer* 和 *les agriculteurs* (见前面关于“富余度”的讨论),无疑会改善模型的质量。
- **非局部依存。**我们知道,诸如 n 元语法模型之类的模型都只能捕捉局部的信息(参阅第 3 章),即使翻译模型生成了正确的词集合,如果词之间存在长距离依存(long-distance dependency),模型将不能生成正确的句子。所以(Brown et al. 1992a)中对句子进行了预处理,减少了长距离依存的数目,例如英文句子 *is she a mathematician* 经过预处理转换为 *she is a mathematician*。
- **词态变化。**模型中具备词态变化的词被区分为不同的词。例如,法语单词 *diriger* 有 39 种词态变化,在不同的上下文中将被翻译为 *to conduct* 或者 *to direct*,而在模型中则需要单独学习每种对应关系。
- **数据稀疏问题。**因为模型参数都是通过训练数据计算得到的,没有借助于其他任何相关词语资源,所以如果数据过少,估计出来的参数可能误差比较大,很难估计出一些不常用单词所具备的真实特征。基于这种原因,Brown et al. (1990) 在评测中将那些含有稀有单词的句子都排除在外了。

总之,上面介绍的噪声信道模型的主要问题在于很少考虑自然语言领域本身的一些知识,这也是 Brown et al. (1990;1993) 中所讨论的问题。以 Brown et al. (1992a) 的工作为标志,之后的统计翻译模型的中心工作在于如何把语言中内在的语言规律性知识融入到模型中。

同时,一些非语言学的模型在词对齐方面被证明也是相当有效的,例如 Brown et al. (1993)。但是,就本章所做的讨论而言,当这些方法应用到机器翻译中时,效果似乎并不好。

13.4 深入阅读

读者如果希望了解关于机器翻译的更多知识,建议阅读 Knight (1997) 的综述文章。如果对高效的解码算法感兴趣,可以参考 Wu (1996), Wang and Waibel (1997), Nießén et al. (1998)。Alshawi et al. (1997), Wang and Waibel (1998), Wu and Wong (1998) 则试图用统计转换的方法代替原始的词对词的直接翻译方法(参见图 13.1), Knight and Hatzivassiloglou (1995) 提出了统计生成算法。

另外一类和噪声信道模型不同的机器翻译方法是基于实例的翻译算法。所谓的基于实例的机器翻译,就是利用源语言和目标语言之间的匹配模板进行翻译。如果对齐语料库中存在精确的匹配,那么遇到相同的句子只需要复制训练数据中的翻译结果,否则需要把原始翻译结果修正为合适的结果。有关基于实例的机器翻译可以参阅 Nagao(1984)和 Sato(1992)。

词对齐的另外一个目的是对含有未登录词的源文的翻译。但是,即使我们从对齐语料库中自动获取词语信息,但是在新文档中仍会出现新的未登录词,特别是名称词。当源语言和目标语言的书写系统完全不同时,例如日文和英文,这类问题会很明显,因为不能采用逐字对应的方法。Knight and Graehl(1997)构建了一个音译系统,用于对应源语言和目标语言之间的专有名词。因为罗马字符在音译为其他字符集合(比如希腊字符集)时有规律可循,所以原始的罗马字符形式能够被完全恢复。

寻找对齐词的过程在更高意义上是机器学习中的知识获取,Knight(1995)在机器学习层面对这个问题做了相当好的介绍。

Brown et al.(1991b)和 Gale et al.(1992d)以及本书 7.2.2 节都对利用对齐问题解决词义消歧问题进行了描述。Shenotov(1993)给出了利用对齐文本协助修改翻译文献的例子,书中一开始的对齐例子来自 Gale and Church(1993),其中并没有讨论词对齐的问题。并且,我们注意到,其实源语言和目标语言文本都是翻译自原始的德文文本的。万维网上给出了一个对齐法文和英文版本的加拿大议会文本的例子,同时也提供了一个搜索界面。

句珠(bead)的概念最早由 Brown et al.(1991c)引入,双文本(bitext)的概念是由 Harris(1988)首先提出的,双语映射(bitext map)的概念则来自于 Melamed(1997a)。在基于信号处理方法的对齐文本上,Melamed(1997a)和 Chang and Chen(1997)做了进一步的研究工作。最近万维网上给出了一些对齐系统的评估结果,而尤其令人感兴趣的是,在具有不同对齐难度的平行语料上,它们的结果相差很大。

13.5 习题

- 13.1 [★] 在我们熟悉的两种语言中,举出一个例子来说明基于长度的句子对齐算法的缺点,即如果源语言和目标语言中有一个短句和长句对应,那么这种对齐算法的性能并不是很好。
- 13.2 [★] Gale and Church(1993)认为基于字符的句子长度相对于基于词的方法更好,因为基于词的方法的方差更大,请读者阐述自己的看法,并说明理由。
- 13.3 [★] 前面描述的点阵图实际上并不完全正确,因为相对于主对角线,它们并不对称,证明该论点的正确性,给出理由。
- 13.4 [★★] 给定输入的法语句以后,统计翻译模型的任务就是确定与其对应的英文句子。那么为什么利用语言模型而不是直接估计 $P(e|f)$ 的值呢? 如果我们依赖于 $P(e|f)$,那么那些不满足语法结构的法语句会出现什么情况呢? 如果我们不考虑法语的语法,而仅仅依赖于 $P(f|e)$,会出现什么情况呢? 这些问题在(Brown et al. 1993:265)中有回答。
- 13.5 [★] 翻译概率和“富余度”概率只能告诉我们生成哪个词,而不能确定它们应放在哪

里,为什么在解码后的大部分词序都是正确的?试说明其原因。

- 13.6 [★★] Viterbi 翻译(Viterbi translation)定义为满足最大似然概率要求的对齐结果就是翻译结果,在翻译模型中不叠加所有的对齐方式的概率,如公式(13.5)所示。说明 Viterbi 翻译和根据公式(13.5)的最优翻译有什么不同?
- 13.7 [★★] 构建一个训练样本集合,采用 EM 算法进行训练,要求算法至少迭代两次。
- 13.8 [★★] 从机器翻译的角度出发, n -gram 语法模型对短句子的描述结果很好。但是,随着句子长度的增加,可能存在很多不同的方式将这些词排列成语法上合理的句子。试寻找两个词集合,分别包含 4 个和 10 个英语单词,每个集合中的单词都可以生成两种语义不同的并且满足语法要求的英文单句。

第 14 章 聚 类

聚类算法的目标是将一组对象划分成若干组或类别,简单地讲就是相似元素同组、相异元素不同组的划分过程。以图 14.1 为例,图中用树图形式描述了 Brown 语料库中 22 个出现频率最高的词的聚类情况,树叶节点表示 22 个英文单词,各个词之间的相似度可以通过它们共同的祖先的高度来度量。共同祖先如果在根节点,则表明两个单词之间的相似度最低,图中每个树节点都由两个子节点聚类而成。例如,单词 *in* 和 *on*, *with* 和 *for* 都属于树图中的同一个子词类,这些子词类进一步合并为语义概念上相对更广泛的词类。这些词类也都表示成了树节点,距离根节点越近,说明其词类越大。子类之间的相似度和它们所处的树节点的高度成反比,树节点和根节点距离越近,其子类之间的相似度越小。如图 14.1 所示,相似度最大的两类个为 *in* 和 *on*,位于树的最底端,相似度最小的类存在于单词 *be* 和其他 21 个单词组成的类别集合之间。

聚类对象的描述需要建立数据表示模型(data representation model)。数据表示模型定义了聚类对象的特征表示形式,不同的对象可以有相同的特征表达式,传统的集合显然不能表示这个概念,因为元素的惟一性假设是集合论的基本前提。因此,聚类算法需要定义在包(bag)的概念之上。与集合不同,包允许相同的元素存在,读者务必区分这两个概念。

但是何谓“相似性”?在 Brown 语料库中,把每个单词的上下文,即左右相邻的词都做了标记并记录下来,由此 Firth 曾经提出过一个词的分类方法。他认为词的上下文提供了足够的信息,用来判断当前词的类别。与第 5 章的重点不同,本章我们希望通过这种记录信息,即词的相邻上下文模式,来判断断词之间的相似度,从而实现聚类操作。按照这个原则,两个词之间的相似性定义为它们相邻上下文模式的重叠程度。例如,单词 *in* 和 *on* 的相似度很高,因为它们的上下文的模式很类似,两者同是介词,后面都跟随冠词或者其他名词短语;单词 *is* 和 *he* 之间的相似度很低,因为语法功能不同,所以它们相邻的上下文几乎没有相同之处。如果假设每个单词为一类,聚类过程就是不断地合并最近似模式类而形成新类的迭代过程。

在统计自然语言处理中,聚类算法有两个重要的用途,第一个重要用途是用于试探性数据分析(exploratory data analysis, EDA)。例如,我们知道,即使一个人对英文一知半解,也能通过对图 14.1 的观察分析,对英文单词的词性有大致的了解,并且会使得后续分析更加简单;对于特定的自然语言处理任务,可以利用类似图 14.1 的图表评估英文中不同词性之间的相似性,即用类别之间的距离表示,当然我们必须假定已经标注了正确的词性。前面我们定义了了在图 14.1 中计算类别距离的方法。相邻上下文模式本身也存在缺点,聚类方法可以说明这一点。例如,在这种方法下,介词有很好的聚类性,在图 14.1 中它们都聚集在一个类里。但是有一些词,诸如 *this* 和 *the*,并没有和语法功能相似的词划分到一个类别中。

对于任何处理“量化”数据的应用来说,试探性数据分析(EDA)是非常重要的方法。当我们面临一个新问题,并且希望建立一个概率模型或者仅仅是为了理解现象的基本特性时,这是一个首要步骤。如果我们盲目地开始操作数据,而不对数据对象做任何事先的理解分析,最终结果常常会不如人意。同时,聚类是试探性数据分析的重要手段之一,尤其是在统计自然语言处理中,因为语言学对象缺乏直接的可视化图景。而其他诸如处理数值数据或者地理数据的

领域,很容易将它们的数据可视化。例如,在研究一种传染病的流行情况时,可以在地图中做上相应的标记。一般来说,可视化数据有助于我们避免对数据做出错误的假定。

在二维平面(例如书本的页面)上显示一系列对象时,还可以使用其他众所周知的技术,参见 14.3 节。聚类方法只是很多试探性数据分析方法中的一种,但是它有其独有的优势,即可以把数据表示成丰富的层次结构,从而可能使应用程序更容易处理数据。可视化数据表示方法实现起来很复杂,不仅需要考虑如何显示的问题,而且可视化环境下对相邻数据的表述方法不利于对数据本质特性的理解。

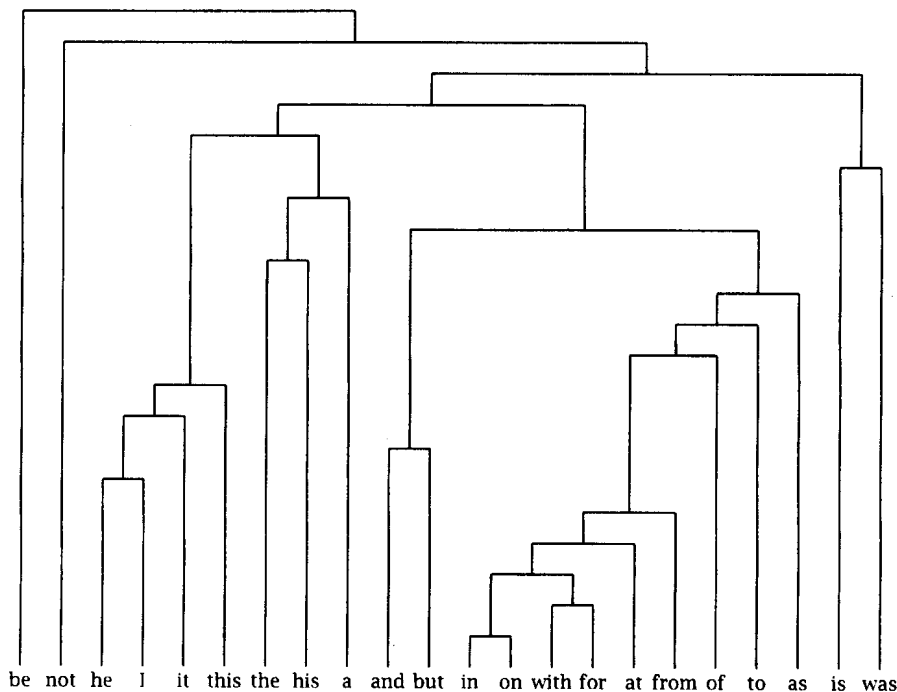


图 14.1 22 个常用英文单词的单连通聚类(single-link clustering)树图

聚类的第二个重要用途就是概念的一般化(*generalization*),在 6.1 节中引入了等价类的概念,但是聚类方法则不同,后者具备某种预先定义好的分类数。

例如,在机器翻译领域,以法翻英为例,如果名词 *Friday* 前的介词未知,当前算法的目的就是要确定这个介词。假设我们手头有一些英文的文本数据,文本中包含一些短语,例如 *on Sunday*, *on Monday*, *on Thursday*,但是没有类似 *on Friday* 的短语,显然我们能推断出 *on Friday* 的正确用法。但是如何让机器去完成这件事呢?首先,在训练集中把英语名词按照语法和语义聚类,那么表示星期的名词会被聚到同一类中,因为它们具备相同的上下文模式,例如 *until day-of-the-week*, *last day-of-the-week*, *day-of-the-week morning* 等,这些属于同类的元素有一个基本特性,即元素之间的可互换性,一个元素的上下文环境同样适用于同类中的其他元素。所以,算法可以从 *on Sunday*, *on Monday*, *on Thursday* 这些模式推断出 *on Friday* 的正确性。在一定程度上讲,聚类过程是一个学习过程,通过聚类可以泛化一些概念。在上面的例子中我们形象地说明了这一点。

分类算法(*classification*)是另外一个类别划分方法,我们在第 16 章中将重点讨论这个问

题。聚类算法和分类算法的区别在于:分类算法是一个有监督的学习过程,它需要对标注数据集进行训练;聚类算法则不需要“教师”的指导,不需要提供训练数据,倾向于数据的自然划分,因此被称为无监督的学习或者自动学习。例如,正是基于介词、冠词和代词的不同上下文模型才得出图 14.1 中的树图结构,而不是依照任何事先定义好的类别。所以聚类也被称为自动或者非监督分类,但是为了避免混淆,我们一般不采用这种称法。

聚类算法很多,但按其方法基本上可以分为两类,层级聚类(hierarchical clustering)和非层级聚类(non-hierarchical clustering 或者 flat clustering)。在非层级聚类中类别结构很简单,并且类别之间的关系没有层级聚类结构清晰。大多数非层级聚类算法是一个迭代过程,算法首先要初始聚类,然后通过不断地迭代重新分配样本数据的类别。

在层级聚类中,每个节点都是父类的一个类(根节点除外),因此聚类可以表现为树图的形式,图 14.1 是层级聚类的一个例子,我们可以看出其中的树叶节点代表了初始的样本,树的内部节点代表了所聚的类,其所有的子节点都是类别中的元素。

聚类算法根据其聚类方法不同,可以分为“软”聚类(soft clustering)和“硬”聚类(hard clustering)。“硬”聚类是指每个样本只能属于一个聚类集合,而“软”聚类中的一个对象可以同时属于几个聚类集合,但是属于各个类别的概率不同,我们用 $P(c_j | x_i)$ 表示对象 x_i 属于类别 c_j 的概率 $P(\cdot | x_i)$,例如样本属于类别 1 的概率是 0.75,属于类别 2 的概率是 0.25,说明该样本更有可能属于类别 1。在向量空间模型中,样本点和类别中心的相似度可以作为这种概率值。在向量空间中,设聚类 c 中含有 M 个样本点,它的中心点同时也被称为中心或者质心点,计算公式如下:

$$\bar{\mu} = \frac{1}{M} \sum_{\vec{x} \in c} \vec{x} \quad (14.1)$$

换言之,向量 $\bar{\mu}$ 是该类别中所有对象向量的均值,公式中的 M 为元素个数。

层级聚类方法通常都是“硬”聚类,在非层级聚类中,“软”聚类和“硬”聚类都有可能,但实际上很多“软”聚类模型都假设对象只属于一个类。它和“硬”聚类的不同之处是,它的结果具有一定的不确定性。当然,很多样本可以同时属于几个模式类别,例如单词 *book* 可以按句法和语义混合聚类,按照语义来理解,它是一个物体,但在语法概念中属于“名词”的范畴,这属于析取聚类(disjunctive clustering)讨论的内容,不是本章重点,有兴趣的读者可以参阅 Saund(1994)。

绝大多数的聚类算法都有前提假设的限制,比如“硬”聚类必须将样本只划分到一个类中。在自然语言处理中,歧义现象很普遍,比如词的兼类,单词 *play* 可以同时充当名词和动词, *fast* 作为形容词和副词都有可能。语法成分越大,其语法现象越复杂,名词性从句可以具有名词或者动词的语法功能,在第 7 章中探讨了这类词义消歧问题。所以,如果采用“硬”聚类方法解决这类问题,通常还需要额外定义类别,比如对于单词 *play*,则需要针对这类单词单独定义一个类别,即同时具有名词和动词特征类别。“软”聚类在解决这一类自然语言处理问题时体现出了它的优越性。例如,通过“软”聚类,单词 *play* 可同时以一定概率属于名词和动词。

本章下面的内容将详细讨论层级和非层级聚类方法,介绍它们在统计自然语言处理中的具体应用。在表 14.1 中,我们已经对它们各自的特点做了一个简短说明,读者可以依照词表判断采用什么聚类方法。

读者如果想进一步了解两者的区别,可以参阅 Kaufman and Rousseeuw(1990),其中对层级

和非层级聚类的优劣进行了细致的讨论。为了方便读者的阅读,首先我们在表 14.2 列出了本章所采用的数学符号及其代表的具体含义。

表 14.1 不同的聚类算法的对比

层级聚类	非层级聚类
■ 适合详细的数据分析过程	■ 在大训练数据条件下或者要求算法效率时非层级聚类更有优势
■ 相对非层级聚类能提供更多的数据信息	■ k 平均聚类算法概念简单,适合处理新问题
■ 不存在最优的聚类算法,现有算法只能证明它对某个应用是最优的	■ k 平均算法适用的数据表达方式有一定的限制,因为它定义在欧氏空间上
■ 算法效率比非层级聚类低,假设集合有 n 个对象,最少也要计算和更新 $n \times n$ 的系数矩阵	■ EM 算法是一种有效的聚类方法,适用于任何定义复杂的概率模型

表 14.2 符 号 表

符号	含义
$X = \{x_1, \dots, x_n\}$	大小为 n 的聚类的数据集合
$C = \{C_1, \dots, C_j, \dots, C_k\}$	类别集合,大小为 k
$\mathcal{P}(X)$	X 的幂集
$\text{sim}(\cdot, \cdot)$	相似度函数
$S(\cdot)$	平均连通相似度函数
m	向量空间维数 $\mathbb{R}^m$
M_j	聚类集合 c_j 中的点数
$\vec{S}(c_j)$	聚类集合 c_j 的向量和
N	训练语料的总词次数
$w_{i, \dots, j}$	训练语料中的字符串(i, j 为下标)
$\pi(\cdot)$	样本的类别映射函数
$C(W^1 W^2)$	子串 $W^1 W^2$ 出现的频度
$C(c_1 c_2)$	满足条件 $\pi(W^1) = c_1, \pi(W^2) = c_2$ 的子串 $W^1 W^2$ 的出现次数
$\vec{\mu}_j$	聚类集合 c_j 的中心
Σ_j	聚类集合 c_j 的协方差矩阵

14.1 层级聚类

层级聚类算法分为自底向上(bottom-up)和自顶向下(top-down)两种。自底向上的方法又称为合并聚类,实质上是一种贪心算法。图 14.2 给出了自底向上的一般算法。每个对象都被初始化为一个类别集合(3,4),然后判断最相似的两个聚类(8),合并为一个集合(9),当只存在一个包含所有对象的类时算法中止。

相反,自顶向下方法开始以全体对象作为初始集合,然后按照组内最大化相似度(8)的方法将它们聚集到不同的组(类别)中。我们已经知道,聚类算法的聚类过程依赖于相似度的具体定义,因为合并的是具有最大相似度的类。所以有些人提出用距离 d 的概念表示类间的相似度,距离和相似度成反比,例如 $\text{sim}(x, y) = 1/(1 + d(x, y))$ 。

```

1  Given: a set  $X = \{x_1, \dots, x_n\}$  of objects
2      a function  $\text{sim}: \mathcal{P}(X) \times \mathcal{P}(X) \rightarrow \mathbb{R}$ 
3  for  $i := 1$  to  $n$  do
4       $c_i := \{x_i\}$  end
5   $C := \{c_1, \dots, c_n\}$ 
6   $j := n + 1$ 
7  while  $|C| > 1$ 
8       $(c_{n_1}, c_{n_2}) := \arg \max_{(c_u, c_v) \in C \times C} \text{sim}(c_u, c_v)$ 
9       $c_j := c_{n_1} \cup c_{n_2}$ 
10      $C := C \setminus \{c_{n_1}, c_{n_2}\} \cup \{c_j\}$ 
11      $j := j + 1$ 

```

图 14.2 自底向上层级聚类算法

图 14.3 中给出了自顶向下的一般算法,也称为分割聚类(divisive clustering),Jain and Dubes (1988:57)。与自底向上算法类似,它也是一种贪心算法。开始时算法中只有一个包含所有样本数据的类别集合(4),因为相似度大的对象之间的内聚度也比较大,所以算法的每一个迭代步骤都是分割最小内聚度(7)的类别集合(8)。例如,如果聚类集合中元素的特征表示方式相同,无疑这个集合具有最大的内聚度。

```

1  Given: a set  $X = \{x_1, \dots, x_n\}$  of objects
2      a function  $\text{coh}: \mathcal{P}(X) \rightarrow \mathbb{R}$ 
3      a function  $\text{split}: \mathcal{P}(X) \rightarrow \mathcal{P}(X) \times \mathcal{P}(X)$ 
4   $C := \{X\}$  ( $= \{c_1\}$ )
5   $j := 1$ 
6  while  $\exists c_i \in C$  s.t.  $|c_i| > 1$ 
7       $c_u := \arg \min_{c_v \in C} \text{coh}(c_v)$ 
8       $(c_{j+1}, c_{j+2}) := \text{split}(c_u)$ 
9       $C := C \setminus \{c_u\} \cup \{c_{j+1}, c_{j+2}\}$ 
10      $j := j + 2$ 

```

图 14.3 自顶向下层级聚类算法

但是,应用层级聚类有一个至关重要的假设前提,即相似度函数是单调函数,满足:

单调性

$$\forall c, c', c'' \in S: \min(\text{sim}(c, c'), \text{sim}(c, c'')) \geq \text{sim}(c, c' \cup c'') \quad (14.2)$$

换言之,聚类的合并过程并不增加样本之间的相似性。例如,树图中原来相似度很小的两个对象,经过一系列的合并操作之后,它们之间反而可能成为相似度很大的对象,显然这是错误的,可见定义的相似度函数如果不满足这个条件,那么“层级”这个概念本身就出现了问题。

绝大多数的层级聚类算法的框架都是基于图 14.2 和图 14.3 的,接下来的章节将陆续介绍一些具体的算法实例。

14.1.1 单连通和全连通聚类

表 14.3 简单介绍了三种相似度函数的大概计算原则,它们都是信息检索系统中经常用到的(van Rijsbergen 1979:36ff)。我们知道,相似度大小直接决定了哪两个类进行合并。首先由来自不同类的样本组成样本数据对,然后计算每个样本对中两个样本的相似度,单连通方法将其中相似度的最大值作为类间的相似度。

表 14.3 聚类算法的相似度函数。单连通和全连通只计算聚类集合之间样本的相似度,平均连通聚类计算的是合并后全体元素的相似度

函数	定义
单连通	两个集合间最相似样本之间的相似度
全连通	两个集合间最不相似样本之间的相似度
平均连通	集合内部样本之间的平均相似度

单连通方法的聚类有好的局部一致性(local coherence),因为它的相似度函数是局部定义的。但是,如图 14.6 所示,聚类区域似乎被拉长了,并且显得杂乱无章。单连通聚类为什么会产生这样的效果?我们结合图 14.4 和图 14.5 进行说明,这两个图描述了单连通的聚类过程,在图中我们以二维空间上的点表示聚类对象,以距离表示相似度。图 14.4 中标出了点之间的距离,因为 $a/b, c/d, e/f, g/h$ 是相似度最大的对象对,所以图中上方和下方的点对首先被合并在一起。图 14.5 给出了合并后的结果,观察此图可以判断出接下来的两个步骤,即分别合并图中上方两个类,然后合并下方的两个类,因为 b/c 和 f/g 是最相似的样本对。图 14.6 是单连通聚类方法的最终结果。虽然两个类中的对象有很好的局部一致性,但是从整体来看,单连通对这几个样本点的聚类效果并不佳,例如在图 14.6 中 a/e 之间的相似度显然要比 a/d 大,但是 a/e 分属不同的类集合, a/d 却在同一个类中。

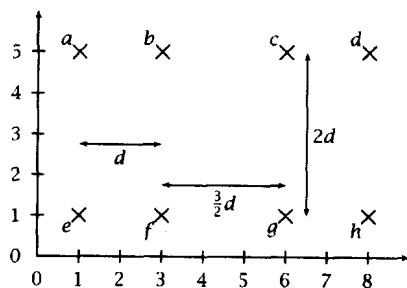


图 14.4 平面点集合

在单连通方法中这种拉长聚类区域的特性称为链式效应(chaining effect),因为我们计算最大相似性的同时没有考虑全局情况。

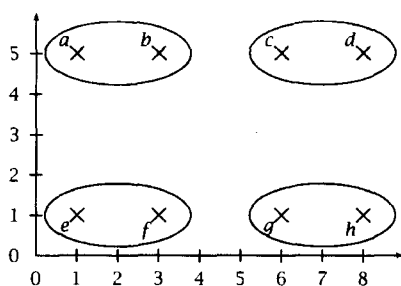


图 14.5 图 14.4 中的点集聚类的中间结果

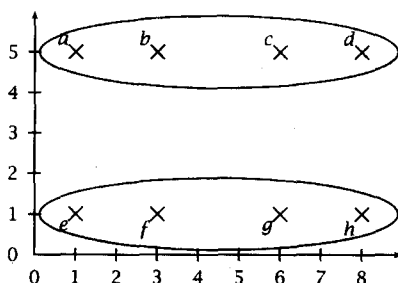


图 14.6 图 14.4 中的点集的单连通聚类示意图

我们可以看出,其实单连通聚类和最小生成树(minimum spanning tree, MST)的方法很类似, MST 每次连接边最短的节点,所以最小生成树是路径之和最短的生成树。MST 同样能构建层级聚类结构,因为每次从 MST 中删除最长的边之后,其结果是两个连通子图,可将其看成是两个聚类子集。

全连通聚类的相似度函数考虑到了全局因素,聚类之间的相似度定义为两个聚类中最不相似样本的相似度,因此它改善了一些单连通聚类方法的缺陷,避免了“拉长”区域的产生。例如,在图 14.5 中全连通聚类会合并左边的两个聚类,然后合并右边的两个聚类,图 14.7 中是全连通聚类的最终结果。因为 a/f 或者 b/e 的相似度比 a/d 小,所以聚类结果比单连通方法显得紧密一些。

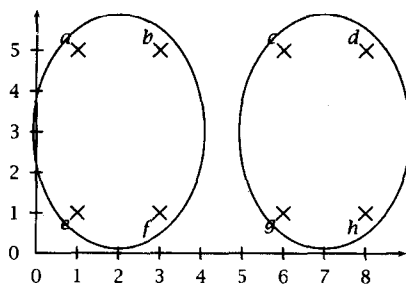


图 14.7 图 14.4 中的点集的全连通聚类示意图

迄今为止,我们一直假定“内部紧密”比“内部松散”的聚类效果好,实际上这源于一个直观感觉,即聚类中的元素都存在中心点,所以人们一般会倾向于结构紧凑的聚类。例如,2.1.9 节介绍的高斯分布模型就符合这一特征。但是这些都是直观判断,本身并不存在理论依据。例如,根据一般的地理常识,夏威夷岛是由一系列直线排列的火山构成的,并且沿直线按照一定的间隔不断地形成新火山。在这种情况下,利用单连通聚类方法无疑是最合适的,因为正是希望利

用它的局部一致性和区域拉长的特性,这样的聚类的结果是沿直线排列的各个火山链,显然全连通聚类算法不能满足要求。所以,不同聚类算法可能产生迥异的结果,但比较而言全连通聚类更适合统计自然语言处理的要求。

全连通聚类的缺点在于它的算法复杂度是 $O(n^3)$ ^①。因为聚类中共有 n 个合并步骤,而对于每个合并步骤又需要 $O(n^2)$,以比较每个聚类对象对,寻找最相似的聚类对象对(n 为聚类对象的数目)。而单连通聚类的算法复杂度为 $O(n^2)$ 。因为一旦 $n \times n$ 相似度矩阵计算好之后,每个合并操作后更新的复杂度为 $O(n)$ 。假设 c_u, c_v 合并为 $c_j c_u \cup c_v$,那么任何一个其他聚类 c_k 和 c_j 的相似度为: $\text{sim}(c_j, c_k) = \max(\text{sim}(c_u, c_k), \text{sim}(c_v, c_k))$ 。也就是说, $n-1$ 个合并步骤中每步最多需要 n 次修改,所以它的复杂度为 $O(n^2)$ 。

除了矩阵表示法以外,单连通和全连通的聚类方法可以用图论理论阐述(Jain and Dubes 1988:64),单连通方法相当于最大连通图问题,全连通方法相当于最大完全图问题。

14.1.2 平均连通聚类

平均连通聚类(group-average agglomerative clustering)是上述两种方法的一个折中方案。与两种方法相比,它既不计算聚类中样本之间的最大相似度(单连通),也不计算最小相似度(全连通),而是将平均相似度作为聚类准则。平均连通聚类可以用来替代全连通聚类,因为它不仅可以避免单连通方法的缺陷,而且它的计算复杂度只有 $O(n^2)$ 。

我们注意到,直接计算平均相似度的算法复杂度为 $O(n^2)$,因此如果算法的每个步骤中都从头计算此值,算法复杂度将为 $O(n^3)$ 。但是,当样本定义在 m 维空间时,相似度量可以采用余弦法。如式(8.40)的定义,所以我们有:

$$\text{sim}(\vec{x}, \vec{y}) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| |\vec{y}|} = \frac{\sum_{i=1}^m x_i \times y_i}{\sqrt{\sum_{i=1}^m x_i^2} \times \sqrt{\sum_{i=1}^m y_i^2}} = \vec{x} \cdot \vec{y} \quad (14.3)$$

公式说明平均相似度的计算可在常量时间内完成,所以平均连通聚类的总的算法复杂度为 $O(n^2)$ 。

我们以 χ 代表待聚类的全部样本集合,每个样本都可以定义为 m 维空间变量。

$$\chi \subseteq \mathbb{R}^m$$

对于任何一个类 $c_j \subseteq \chi$,类 c_j 的平均相似度 S 定义如下,其中因子 $|c_j|(|c_j| - 1)$ 为非零相似度的总数:

$$S(c_j) = \frac{1}{|c_j|(|c_j| - 1)} \sum_{\vec{x} \in c_j} \sum_{\vec{y} \in c_j, \vec{y} \neq \vec{x}} \text{sim}(\vec{x}, \vec{y}) \quad (14.4)$$

设 C 表示当前所有聚类组成的集合,算法的每次迭代都确定两个类集合 c_u 和 c_v ,使 $S(c_u \cup c_v)$ 取值最大(相当于图 14.2 中的步骤 8),这个步骤可以在线性时间内完成(Cutting et al. 1992: 328), C' 表示合并操作后的聚类集合,那么(相当于图 14.2 中的步骤 10):

① “ $O(n^3)$ ”属于算法复杂度中的“大 O ”表示法,这里我们假设读者已熟知这类表示法,绝大多数涉及算法的书籍都有相关的介绍,比如(Cormen et al. 1990)。这类符号表示只表明了与算法有基本依赖关系的特定参数,而忽略了一些常数项。

$$C' = (C - \{c_u, c_v\}) \cup \{c_u \cup c_v\}$$

为了在计算平均相似度时尽量减少其重复计算量,首先计算下式,即聚类中样本的向量和 $\bar{s}(c_j)$:

$$\bar{s}(c_j) = \sum_{\bar{x} \in c_j} \bar{x}$$

可以看出向量和 $\bar{s}(c_j)$ 的定义有两个特点,一是聚类合并后其值很容易更新,只需要把两个原有聚类对象的向量和相加;二是聚类平均相似度 $S(c_j)$ 的计算可以利用 $\bar{s}(c_j)$ 的计算结果,下式给出了它们之间的关系:

$$\begin{aligned} \bar{s}(c_j) \cdot \bar{s}(c_j) &= \sum_{\bar{x} \in c_j} \bar{x} \cdot \bar{s}(c_j) \\ &= \sum_{\bar{x} \in c_j} \sum_{\bar{y} \in c_j} \bar{x} \cdot \bar{y} \\ &= |c_j|(|c_j| - 1)S(c_j) + \sum_{\bar{x} \in c_j} \bar{x} \cdot \bar{x} \quad (14.5) \\ &= |c_j|(|c_j| - 1)S(c_j) + |c_j| \\ \text{所以, } S(c_j) &= \frac{\bar{s}(c_j) \cdot \bar{s}(c_j) - |c_j|}{|c_j|(|c_j| - 1)} \end{aligned}$$

因此,如果两个聚类 c_i 和 c_j 的向量和 $\bar{s}(\cdot)$ 已知,那么它们合并形成的聚类的平均相似度计算公式可以写为:

$$S(c_i \cup c_j) = \frac{(\bar{s}(c_i) + \bar{s}(c_j)) \cdot (\bar{s}(c_i) + \bar{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)} \quad (14.6)$$

可以看出,采用这种计算的平均连通聚类的算法复杂度为 $O(n^2)$ 。在计算过程中,关键步骤是算法首先需要计算所有初始聚类样本之间的相似度,它保证了随后的 n 次合并过程都可以在线性时间内完成。

因此,在处理一些多特征数据集时,该算法体现出了它的有效性。但是,聚类算法的计算时间仍然和向量空间的具体定义有关。到目前为止,还不存在一种通用的、独立于任何数据表示模型的平均聚类方法。

14.1.3 聚类在改进语言模型上的应用

前面几节介绍了几种典型的层级聚类方法,本节将着重介绍聚类算法的具体应用,例如利用聚类算法建立一个更准确的语言模型(language model)。本书前面的章节曾经引入了语言模型的概念,它在语音识别或机器翻译中都有重要应用,例如在语音识别中经常需要处理类似 *President Kennedy* 和 *precedent Kennedy* 的判断问题,因为它们的发声完全相同,但是语言模型可以告诉我们 *President Kennedy* 更有可能。它的推理过程是基于噪声信道模型的(见 2.2.4 节),即要求假设 H 满足 $P(H)$ 和条件概率公式 $P(D|H)$ 的乘积最小,其中 D 为观测到的语音信号。

$$\hat{H} = \arg \max_H P(H|D) = \arg \max_H \frac{P(D|H)P(H)}{P(D)} = \arg \max_H P(D|H)P(H)$$

聚类方法在改进语言模型质量上可以起到重要作用,正如第 6 章所介绍的,在构建语言模型时我们面临的一个困境是语料数据稀疏问题,很多小概率事件往往不能在模型中体现出来,聚类方法可以通过泛化形式,通过在聚类基础上的建模,在一定程度上提高语言模型的准确度,这种方法首先被 Brown et al. (1992c) 使用,取得了很好的实验效果。

语言模型 (language model)

迄今为止,我们讨论的都是二元语法模型,即满足一阶马尔可夫假设的模型,即当前词的出现概率仅和前面的一个词有关。语言模型的评估标准是交叉熵 (cross entropy) 或者混乱度 (perplexity),具体内容可以参见 2.2.8 节,它们衡量是模型的不确定性。这里我们的目的是寻找一个聚类函数 π 将词聚类,相对于简单的二元语法模型,我们希望通过使用聚类算法减少它的混乱度。

首先计算训练语料 $L = w_1 w_2 \cdots w_N$ 的交叉熵,假设聚类函数为 π ,它将每个词映射到某个聚类中,训练语料满足一阶马尔可夫假设前提,那么它的计算公式为:

$$H(L, \pi) = -\frac{1}{N} \log P(w_1, \dots, w_N) \quad (14.7)$$

$$\approx -\frac{1}{N-1} \log \prod_{i=2}^N P(w_i | w_{i-1}) \quad (14.8)$$

$$\approx -\frac{1}{N-1} \sum_{w^1 w^2} C(w^1 w^2) \log P(w^2 | w^1) \quad (14.9)$$

然后我们假定当前聚类集合 c_2 中的一个词的出现仅仅依赖于前面一个词的聚类类别 c_1 ^①,式(14.9)变为:

$$H(L, \pi) \approx -\frac{1}{N-1} \sum_{w^1 w^2} C(w^1 w^2) \log P(c_2 | c_1) P(w^2 | c_2) \quad (14.10)$$

式(14.10)可简化为:

$$H(L, \pi) \approx - \left[\sum_{w^1 w^2} \frac{C(w^1 w^2)}{N-1} [\log P(w^2 | c_2) + \log P(c_2)] \right] \quad (14.11)$$

$$\begin{aligned} & + \sum_{w^1 w^2} \frac{C(w^1 w^2)}{N-1} [\log P(c_2 | c_1) - \log P(c_2)] \Big] \\ = & - \left[\sum_{w^2} \frac{\sum_{w^1} C(w^1 w^2)}{N-1} \log P(w^2 | c_2) P(c_2) \right. \\ & \left. + \sum_{c_1 c_2} \frac{C(c_1 c_2)}{N-1} \log \frac{P(c_2 | c_1)}{P(c_2)} \right] \end{aligned} \quad (14.12)$$

① 我们可以注意到,除了用单词所属的聚类类别代替了词性之外,这个公式和第 10 章中的词性标注公式有很多相似之处。

$$\approx - \left[\sum_w P(w) \log P(w) + \sum_{c_1 c_2} P(c_1 c_2) \log \frac{P(c_1 c_2)}{P(c_1)P(c_2)} \right] \quad (14.13)$$

$$= H(w) - I(c_1; c_2) \quad (14.14)$$

在式(14.13)的推导中隐含使用了两个近似等式,即 $\frac{\sum w^1 C(w^1 w^2)}{N-1} \approx P(w^2)$ 以及 $\frac{C(c_1 c_2)}{N-1} \approx P(c_1 c_2)$, (当 n 的值很大时)。此外因为 $\pi(w_2) = c_2$, 所以 $P(w_2 | c_2)P(c_2) = P(w_2 | c_2) = P(w_2)$ 。

式(14.14)表明可以通过聚类函数 π 的合理构建最小化交叉熵,即,使互信息 $I(c_1; c_2)$ 最大化,所以我们应该选择使其互信息最大的聚类集合来得到最优的语言模型。

聚类

假设我们采用的是图 14.2 中描述的自底向上的聚类方法,并定义一个如下的互信息损失函数,即最大化相邻类之间的互信息,作为相邻类的合并准则:

$$\text{MI-loss}(c_i, c_j) = \sum_{c_k \in C \setminus \{c_i, c_j\}} I(c_k; c_i) + I(c_k; c_j) - I(c_k; c_i \cup c_j) \quad (14.15)$$

在每一个聚类步骤中,按照信息损失最小原则选定两个聚类进行合并操作,实际上相当于基于下面的聚类选择准则进行选择:

$$(c_{n_1}, c_{n_2}) := \arg \min_{(c_i, c_j) \in C \times C} \text{MI-loss}(c_i, c_j)$$

聚类过程的终止条件是,聚类集合数目是否达到一个预定义的阈值 k , 比如 1000 (Brown et al. 1992c)。当然,在具体计算过程中可以采取一些技巧以减少计算量。另外,聚类还是一种贪心算法,不保证全局最优,所以在词聚类完成之后,可以在各类之间移动一些词,以提高其聚类效果。有兴趣的读者可以参阅 (Brown et al. 1992c), 它共找出了 1000 个英文词类,下面是其中的三个类:

- plan, letter, request, memo, case, question, charge, statement, draft
- day, year, week, month, quarter, half
- evaluation, assessment, analysis, understanding, opinion, conversation, discussion

我们看出其中每一类都有其特定的语法或者语义含义,比如上面的第二类都是表示时间段的名词。

但是在计算了混乱度之后,会发现语言模型并没有明显的改善,原来基于词的语言模型的混乱度为 244 (Brown et al. 1992c:476), 词聚类后为 277, 但是两者的插值模型为 236 (关于插值的介绍参见 6.3.1 节), 优于原有的词模型。

实际上聚类过程和利用聚类的推导在语言模型中是统一的,聚类的指导准则 $H(L, \pi) = H(w) - I(c_1; c_2)$, 同时可以作为模型的衡量标准,它的值和模型好坏成反比。有一些研究者把聚类和语言建模分开处理,但是我们更倾向于把聚类和基于聚类的推理结合在一起,因为它保持了聚类和泛化之间的一致性,从而具备了最优特性。

14.1.4 自顶向下聚类

图 14.3 中描述了自顶向下的层级聚类法,算法每次都分割内聚度最小的类。表 14.3 中的相似度函数同样可以作为自顶向下聚类中的内聚度衡量标准。按照单连通方法,聚类的内

聚度相当于其最小生成树中的最小相似度;按照全连通方法,内聚度是两个聚类之间最不相似的两个样本之间的距离;按照平均连通聚类方法,则相当于聚类样本的平均相似度。而这三种都可以作为自顶向下迭代步骤中的最小内聚度的计算方法。

但是,自顶向下聚类操作内部还需要有一个聚类过程,所以它不经常被采用。分割过程同样也需要寻找两个子聚类,但是其聚类的方法不受限制,可以是任何一种聚类算法,比如前面所述的自底向上的方法或者非层级聚类法。

自顶向下的层级聚类法在解决某些问题时也能体现出其优越性。例如概率分布的聚类,常用的概率之间的相似度函数为 KL 距离(Kullback-Leibler Divergence),2.2.5 节中给出了其计算公式:

$$D(p \parallel q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} \quad (14.16)$$

当 $p(x) > 0, q(x) = 0$ 时,函数无定义。所以,当聚类样本的概率分布中有很多零概率事件时,显然不能应用上面的公式,从而不可能计算相似度矩阵,所以出现这种情况时不能采用自底向上聚类方法。

例如,名词分布的聚类问题(Pereira et al. 1993)用名词作为特定动词宾语的概率大小刻画名词的概率分布,设定 $q_n(v)$ 表示给定宾语名词 n 时谓语动词 v 的相对出现频度。例如,名词为 *apple*,动词为 *eat* 时, $q_n(v) = 0.2$,表示语料数据中所有 *apple* 中有 1/5 是作为 *eat* 的宾语出现的。

任何一个名词只能作为有限谓语动词的宾语,这样就会出现前面提到的在概率分布函数中存在很多概率为零的事件的情况,因此自底向上的聚类方法不能用来解决此问题。但是,自顶向下聚类的方法可以解决这个名词分布聚类问题。首先将聚类中所有样本的概率和做归一化处理,作为聚类中心点,计算其样本和聚类中心点之间的 KL 距离,因为聚类中心点的概率分布中零点很少,所以避免了上述情况的出现,详细的算法过程可参阅(Pereira et al. 1993)。

14.2 非层级聚类

非层级聚类通常都有一个初始划分假设,任何样本都可以作为种子(seed),即初始的聚类中心点,然后在此基础上不断地更新聚类划分。层级聚类采用的是所谓的一次通过法(single-pass method),只需要扫描所有聚类样本数据一次。绝大多数非层级聚类都需要几次迭代,每次迭代都有可能将样本数据再分配(reallocate),把样本从一个类重新分配到另外一个类中。聚类样本的重新分配实际上是其在不同类之间的移动,例如在 14.1.3 节中,语言模型通过聚类对象的移动来改善全局互信息。

如果非层级聚类包含了多次迭代过程,那么有必要定义迭代过程的停止准则函数。停止准则函数是一种适合度或者聚类质量的度量,前面提到的平均相似度和互信息都满足这个要求。稍后我们将引入另外一种很重要的聚类适合度,它相当于在聚类模型给定的条件下对数据可能性的一种描述。但是,无论我们用哪种方法终止迭代过程,基本原则都是确保每次迭代都改善聚类效果,当改善的幅度减缓时就可以停止迭代过程。

引入聚类适合度(goodness)的同时又带来了另外一个问题:如何确定聚类过程中的类别数目?先验知识有时能提供类似的信息,例如词性类别的聚类。但是,大多数情况下不具备这样的先验知识。虽然聚类类别数目和聚类的效果成正比,在上节的语言模型的例子中,显然聚类数越多,作为衡量标准的互信息的值会越大。但是,如果训练数据本身就具备某种特定的自然

划分方式,比如分为 k 类,那么聚类数目由 $k-1$ 变为 k 时,我们可以观察到明显的改进效果。但是,当由 k 变为 $k+1$ 时,其相应的改善程度要小得多。所以,在聚类过程中,可以寻找具备这种特征的 k 作为数据的聚类数。

寻找最优聚类数的一种更合理的方法是**最小描述长度法**(Minimum Description Length, MDL),Cheeseman et al.(1988)在 AUTOCLASS 系统中用到了这种方法^①。它的基本思想是要求聚类适合度(goodness)的定义同时考虑对象如何聚类以及聚类数目如何确定的问题,比如对聚类数目定义一个惩罚度,对高聚类数情况下的适合度进行惩罚。在 MDL 框架下,类和聚类对象都用字编码表示(code word),字编码长度单位是比特。类别越多,类别编码所需比特数就越多,与此同时样本所用比特数会变少,因为样本的编码可以利用所属类别和它们之间的差异值表示。所以,定义损失函数时同时要考虑聚类对象和聚类数目的编码长度,那么如果迭代过程最小化此损失函数,就可以同时获得聚类数目和样本所属的类别。

直观上感觉层级聚类法不需要考虑聚类数目的因素,但是层级聚类并不表明聚类结果固定不变,因为通过树图的剪枝可以得到不同的聚类形式。通常,为了获得一个有实际用途,比如固定聚类数目的聚类结果,剪枝操作必不可少。从这个角度上讲,层级聚类和非层级聚类并没有本质上的不同,非层级聚类的优点在于其算法的效率高。

我们将重点介绍两种非层级聚类的方法, k 平均和 EM 算法。 k 平均算法可能是最简单的聚类算法,虽然有其局限性,但却因为适用面广、效率高而得到广泛应用。EM 算法是一种算法的基本框架,可以解决很多问题,本章首先给出应用 EM 算法的一个具体实现,随后探讨它的一般形式。EM 算法在统计自然语言处理中有很广泛的应用,例如向内-向外算法(inside-outside algorithm)和前向-后向算法(forward-backward algorithm)。

14.2.1 k 平均算法

k 平均算法是一种“硬”聚类算法。开始时需要设置初始的聚类中心,将样本类别判定为距某聚类中心最近的类别,当所有的数据对象都按照这个原则确定其类别之后,算法重新计算每个聚类中心或其样本均值 $\vec{u}$, $\vec{u} = (1/|c_j|) \sum_{i \in c_j} \vec{x}_i$,见图 14.8(假设样本数据之间的距离为欧氏距离),直到迭代结束。

```

1  Given: a set  $X = \{\vec{x}_1, \dots, \vec{x}_n\} \subseteq \mathbb{R}^m$ 
2      a distance measure  $d: \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ 
3      a function for computing the mean  $\mu: P(\mathbb{R}) \rightarrow \mathbb{R}^m$ 
4  Select  $k$  initial centers  $\vec{f}_1, \dots, \vec{f}_k$ 
5  while stopping criterion is not true do
6      for all clusters  $c_j$  do
7           $c_j = \{\vec{x}_i \mid \forall \vec{f}_l d(\vec{x}_i, \vec{f}_l) \leq d(\vec{x}_i, \vec{f}_j)\}$ 
8      end
9      for all means  $\vec{f}_j$  do
10          $\vec{f}_j = \mu(c_j)$ 
11     end
12 end
```

图 14.8 k 平均聚类算法

^① AUTOCLASS 可以从本书配套网站下载得到。

k 平均算法对“脏”数据很敏感,例如在 k 平均算法中,用欧氏距离作为样本距离度量时经常会出现某类中只有一个样本数据的情况,因为这个孤立样本离其他的样本点都很远,而且它很可能是噪声数据。一个利用 L_1 范式(参见 8.5.2 节)的 k 平均变种可以避免这个问题。

$$L_1(\vec{x}, \vec{y}) = \sum_i |x_i - y_i|$$

这个公式的好处在于其对外部孤立点不敏感,所以 L_1 空间的聚类倾向于相同大小的聚类集合,并且采用称为 medoid 的点代替上面提到的聚类中心。两者之间的区别在于 medoid 是最接近中心的实际样本点,而原来的聚类中心是样本均值。

k 平均算法的迭代次数为常数,因此它的算法复杂度为 $O(n)$ 。图 14.9 给出了一个 k 平均算法的具体迭代过程。首先,样本对象按其与初始样本中心的距离聚类,图 14.9 的例子是一个特例,它只需迭代一次,因为其聚类中心点不再改变。

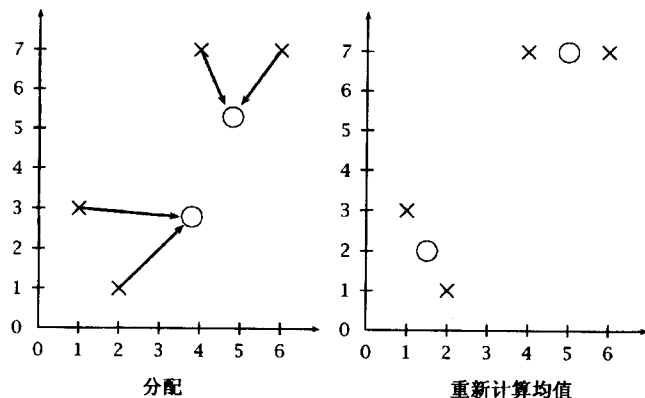


图 14.9 k 平均算法的一次迭代过程,图中圆圈表示聚类中心点

到目前为止我们一直没有讨论样本数据与几个聚类中心点距离都相等的情况。针对这种情况,有两种可以采取的措施,一是随意将其判定为某个类别,二是稍微对该样本数据加入一些扰动信息,从而避免“等距”情况的出现。

下面是 k 平均聚类算法在自然语言处理中的一个应用,首先从纽约时报语料库中挑选出 20 个单词(见第 5 章),它们是:

Barbara, Edward, Gov, Mary, NFL, Reds, Scott, Sox, ballot, finance, inning, payments, profit, polls, quarterback, researchers, science, score, scored, seats

表 14.4 是这 20 个词的 k 平均聚类的结果($k=5$),数据的表示方法是词共现计数向量空间,其具体定义参阅第 8 章的内容。如表 14.4 所示,前四个类别分别对应于: *government*, *finance*, *sports* 和 *research*,最后一个类别对应于姓名。可见词的聚类使我们更容易理解单词的属性及其它它们之间的内在联系。

k 平均算法中的初始聚类中心点是随机选取的,但是它们和最终的聚类效果关系并不大,这是因为聚类效果取决于样本数据集合本身的结构。如果样本数据本身定义良好(well-behaved set),那么即使我们初始的聚类中心点的选择不同,最终的聚类结果都是相同的。

当样本数据本身为非定义良好的数据集合(ill-behaved set)时,我们可以首先利用层级聚类算法在样本的一个子集上聚类,确定一些合理的 k 平均算法初始聚类中心点,这也是 Buck-

shot 算法的基本思想。Buckshot 算法首先在原样本集合的一个随机取样子集上应用平均连通聚类法(GAAC),子集合大小为其父集大小的平方根,所以 GACC 的算法复杂度为 $O(n)$,随后在这个基础上利用 k 平均算法聚类。

表 14.4 k 平均聚类的例子,20个单词用词共现计数向量空间表示,聚类数目为 5,括号中的数字表示单词和聚类中心点之间的距离

聚类	成员
1	ballot (0.28), polls (0.28), Gov (0.30), seats (0.32)
2	profit (0.21), finance (0.21), payments (0.22)
3	NFL (0.36), Reds (0.28), Sox (0.31), inning (0.33), quarterback (0.30), scored (0.30), score (0.33)
4	researchers (0.23), science (0.23)
5	Scott (0.28), Mary (0.27), Barbara (0.27), Edward (0.29)

14.2.2 EM 算法

EM 算法是一种“软”聚类方法,它并不是一个具体可操作的算法,实际上只是提供了解决问题的框架。图 14.10 描述了一个简单的例子,如前所述,假定开始随机选择的聚类中心为 c_1 和 c_2 , k 平均聚类算法只需一次迭代即可得到右图的最终结果。EM 算法则假定样本可能属于每个类别,例如图 14.10 里中图右下方的点可以同时属于 c_1 和 c_2 ,只是属于 c_2 的概率大于其属于 c_1 的概率。EM 算法只有经过了两次迭代才能达到一个稳定的状态。

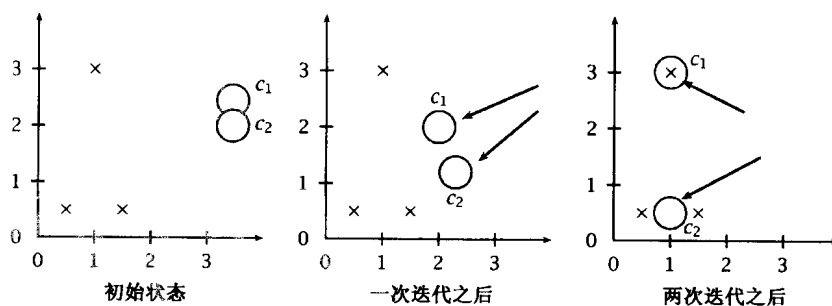


图 14.10 使用 EM 软聚类算法的一个例子

在很多机器学习问题中,除了一部分可以观察到实例数据之外,还有一些隐含变量没有包含在观察数据中,例如 HMM 模型中其内部的状态变化对外部来说是无法直接观测到的。那么如何用这些观测到的数据去预测隐含变量呢?此外还有一种情况,我们知道在模型的参数估计中,如果有足够的定义良好的训练特征,完全可以使用简单的极大似然估计法来估计模型参数。举例来说,如果变量 X 是可观察的,并且概率模型 p 已知, $P(X|p(\Theta))$ 表示概率模型中数据 X 的出现概率, Θ 为参数空间。我们的目标是利用极大似然估计寻找参数空间 Θ 中的一组参数,使该概率值最大。假设概率空间连续平滑,对于模型中的每个参数 θ_i ,可以通过一阶导数的方法求解,即 $\frac{\partial}{\partial \theta_i} \log P(\dots) = 0$,但是得到的往往是一组不易求解的非线性等式,而且有时可能缺乏某些必要特征或者存在“脏”特征。而 EM 算法正是一种解决这类问题的有效的学习方法。

解释 EM 的最好方法是通过实例说明。我们首先介绍一个利用 EM 算法求解高斯混合分布参数的实例(见图 14.10),假设高斯混合分布是由 k 个不同的高斯分布混合生成的;随后引出 EM 算法的一般表述形式,并将其应用于诸如向内-向外算法、前向-后向算法的参数估计问题。

EM 算法估计高斯混合分布参数

在把 EM 算法用于聚类的时候,我们将聚类看成是一个估计混合概率分布的问题。其思想是把观测到的数据看成是由一些隐藏原因生成的。每个原因都对生成过程有自己的独立贡献,但我们只看到了最后的总体情况,而看不到各个原因的单独贡献情况。我们把这个概念通过一对数据来形式化。假设有观测数据 $X = \{\tilde{x}_i\}$, 其中 $\tilde{x}_i = \{x_{i1}, x_{i2}, \dots, x_{im}\}^T$ 简单对应第 i 个数据点的向量。未观测到的数据为 $Z = \{\tilde{z}_i\}$, 其中 $\tilde{z}_i = \{z_{i1}, z_{i2}, \dots, z_{ik}\}^T$ 。如果个体 i 属于第 j 个聚类(即假定由前面提到的隐藏原因生成),则 $z_{ij} = 1$, 否则为 0。

如果我们知道每个聚类的分布情况(或者生成原因),就可以使用 EM 算法来聚类。在估计高斯混合分布时,假设每个聚类都是高斯分布的。EM 算法于是可以确定混合分布的最可能的参数(每个高斯分布的均值和方差)以及每个分布的先验概率(相对重要性或权重)。总的来说,我们假设聚类的数据包含由 k 个高斯分布($n_1 \dots n_k$)生成的 n 个 m 维的个体 $X = \{\tilde{x}_1 \dots \tilde{x}_n\} \subset R^m$ 。

既然我们将混合高斯分布看成是聚类的结果,其中每个高斯分布相当于一个聚类,那么对于每个个体 $\tilde{x}_i$, 我们可以计算概率 $P(w_j | \tilde{x}_i)$, 它表示第 j 个高斯分布生成 $\tilde{x}_i$ 的概率。一个个体可以以不同的置信度同时属于几个聚类。

多元正态(高斯)分布(Multivariate normal distributions): 假设定义在 m 维空间中的高斯分布函数簇,其均值为 $\vec{\mu}_j$, 协方差矩阵为 Σ_j , 它是一个 $m \times m$ 的对称正定可逆矩阵,则其概率密度函数为:

$$n_j(\vec{x}; \vec{\mu}_j, \Sigma_j) = \frac{1}{\sqrt{(2\pi)^m |\Sigma_j|}} \exp \left[-\frac{1}{2} (\vec{x} - \vec{\mu}_j)^T \Sigma_j^{-1} (\vec{x} - \vec{\mu}_j) \right] \quad (14.17)$$

如前所述,假定数据是由 k 个相互独立的高斯分布生成的,我们的目标在于如下形式的极大似然估计, π_j 表示高斯分布的先验概率,满足 $\sum_j \pi_j = 1$ 。

$$\sum_{j=1}^k \pi_j n_j(\vec{x}_j; \vec{\mu}_j, \Sigma_j) \quad (14.18)$$

表 14.5 是一个高斯混合分布的例子,我们假设表 14.4 中的样本数据是由高斯混合分布生成的,每个单词都有其主导类别,例如单词 *ballot* 属于类别 1 的概率要比属于其他类别的概率大得多,但是无论其概率值大小,每个单词都以不同的概率值属于 5 个类别。在评估单词和某个主题的相关程度时,这种“软”聚类的方法能体现出其优越之处。举例说明, *inning* 和 *score* 都是 *sport* 类别中的单词,但是 *inning* 属于 *sport* 的概率为 0.93, *score* 的概率只有 0.65,同时它属于 *government* 的概率为 0.12,说明 *score* 还和其他类别有关。这是一个软聚类应用的好例子。

下面我们讨论如何具体使用 EM 算法估计高斯分布的参数。设 $\theta_j = (\vec{\mu}_j, \Sigma_j, \pi_j)$ 为第 j 个高斯分布的参数,需要估计的参数空间即可表示为: $\Theta = (\theta_1, \dots, \theta_k)^T$, 样本 X 的概率公式为:

$$\begin{aligned} l(X|\Theta) &= \log \prod_{i=1}^n P(\vec{x}_i) = \log \prod_{i=1}^n \sum_{j=1}^k \pi_j n_j(\vec{x}_i; \vec{\mu}_j, \Sigma_j) \\ &= \sum_{i=1}^n \log \sum_{j=1}^k \pi_j n_j(\vec{x}_i; \vec{\mu}_j, \Sigma_j) \end{aligned} \quad (14.19)$$

上式的参数计算利用了极大似然估计原理,它本质上是在满足一定限制条件下的求极值问题,直接计算存在着很大困难,并且公式含有对数,又增加了计算的难度。但是,EM 算法可以通过迭代过程给出其近似解。

表 14.5 高斯混合分布的例子。样本共有 5 个聚类类别,均值 $\bar{\mu}_j$ 即为表 14.4 中的聚类中心点,协方差矩阵 $\Sigma = 0.05 \cdot I, \pi_j = 0.2$ 为均匀分布,在这些值的基础上可以计算样本数据的概率值 $P(w_i | c_j)$

聚类	单词	$P(w_i c_j) = n_j(\bar{x}_i; \bar{\mu}_j, \Sigma_j)$				
		1	2	3	4	5
1	ballot	0.63	0.12	0.04	0.09	0.11
1	polls	0.58	0.11	0.06	0.10	0.14
1	Gov	0.58	0.12	0.03	0.10	0.17
1	seats	0.55	0.14	0.08	0.08	0.15
2	profit	0.11	0.59	0.02	0.14	0.15
2	finance	0.15	0.55	0.01	0.13	0.16
2	payments	0.12	0.66	0.01	0.09	0.11
3	NFL	0.13	0.05	0.58	0.09	0.16
3	Reds	0.05	0.01	0.86	0.02	0.06
3	Sox	0.05	0.01	0.86	0.02	0.06
3	inning	0.03	0.01	0.93	0.01	0.02
3	quarterback	0.06	0.02	0.82	0.03	0.07
3	score	0.12	0.04	0.65	0.06	0.13
3	scored	0.08	0.03	0.79	0.03	0.07
4	researchers	0.08	0.12	0.02	0.68	0.10
4	science	0.12	0.12	0.03	0.54	0.19
5	Scott	0.12	0.12	0.11	0.11	0.54
5	Mary	0.10	0.10	0.05	0.15	0.59
5	Barbara	0.15	0.11	0.04	0.12	0.57
5	Edward	0.16	0.18	0.02	0.12	0.51

EM 算法主要分为两个步骤,估计步骤和最大化步骤。算法重复这两个步骤直至收敛。

估计 (Estimate): 利用当前参数空间 Θ 计算概率模型中隐藏变量的期望值。

最大化 (Maximize): 假设模型中隐藏变量的期望值已知,我们在其基础上计算参数空间 Θ 的极大似然估计。

我们首先为 Θ 给出一个猜测值,并在期望步骤(expectation step)与最大化步骤(maximization step)之间前后迭代,从而打破了这个循环,这个算法因此称为 EM 算法。在期望步骤中,我们计算了隐藏变量 z_j 的期望值。隐藏变量也可以被认为是属于某个聚类的成员概率。接下来我们可以得到一个个体属于某个聚类的可能性。在最大化过程中,我们计算得到给定类成员概率情况下最可能的模型参数。这个过程改进了我们的模型参数估计,因此聚类参数能够更好地反映类中具有较高类成员概率的个体属性。

EM 算法的另一个重要属性是单调性。每经过一次迭代,模型对于给定数据的似然性都会增加,这样就保证了每次迭代都会产生更优的模型参数。但是,如果算法最终落入了局部极值,就没有办法再找到全局最优解。这是与最小平方方法(如 SVD, 参见第 15 章)的一个重要差异,最小平方方法就能保证找到全局最优解。

接下来我们描述一个估计高斯混合分布的 EM 算法。我们采用与 (Dempster et al. 1977), (Mitchell 1997: ch.6) 以及 (Ghahramani 1994) 类似的讨论方式。读者也可以参阅 (Duda and Hart 1973: 193)。

在高斯混合分布模型中,首先初始化参数空间,假设每个高斯分布的协方差矩阵为单位矩阵,高斯分布的先验概率 $\pi_j = 1/k$,这个值又称为聚类权重值。 k 个高斯分布的均值 $\vec{\mu}_i$ 设定为任何值,即在随机选择的样本点上加入干扰数据。

估计步骤:这一步计算参数 h_{ij} ,用 h_{ij} 代表隐藏变量 z_{ij} 的期望值,如果 n_j 生成 $\vec{x}_i$,则 $z_{ij} = 1$,否则 $z_{ij} = 0$,其计算公式如下:

$$h_{ij} = E(z_{ij}|\vec{x}_i; \Theta) = \frac{\pi_j n_j(\vec{x}_i; \Theta)}{\sum_l \pi_l n_l P(\vec{x}_i; \Theta)} \quad (14.20)$$

最大化步骤:给定 h_{ij} 之后,重新估计均值参数空间 Θ 的极大似然估计,计算公式如下 (Duda and Hart 1973:23):

$$\vec{\mu}'_j = \frac{\sum_{i=1}^n h_{ij} \vec{x}_i}{\sum_{i=1}^n h_{ij}} \quad (14.21)$$

$$\Sigma'_j = \frac{\sum_{i=1}^n h_{ij} (\vec{x}_i - \vec{\mu}'_j) (\vec{x}_i - \vec{\mu}'_j)^T}{\sum_{i=1}^n h_{ij}} \quad (14.22)$$

各个类别,即高斯分布模型的先验概率,也必须重新计算:

$$\pi'_j = \frac{\sum_{i=1}^n h_{ij}}{\sum_{j=1}^k \sum_{i=1}^n h_{ij}} = \frac{\sum_{i=1}^n h_{ij}}{n} \quad (14.23)$$

不断地迭代 E 和 M 步骤,重复计算高斯分布的均值、协方差以及其先验概率,直到式(14.19)所表示的概率值不再有显著增加。

EM 算法在统计自然语言中的应用

上面我们介绍了如何应用 EM 算法估计高斯混合分布的参数,但是 EM 算法是一个一般性的学习框架,应用很广泛,可以解决很多具体问题。EM 算法的一般形式是 Dempster et al. (1977:6)提出的,它定义了一个 Q 函数:

$$Q(\Theta|\Theta^k) = E(l((X, Z)|\Theta)|X, \Theta^k)$$

其中参数 z 是隐藏变量, x 是可观察到的数据,相当于前面的向量 $X = \{\vec{x}\}$, Θ 为模型的参数空间,相当于高斯分布的均值、协方差和先验概率。 $l((X, Z)|\Theta)$ 表示观察数据和未观察数据的联合分布函数。EM 算法的一般表述形式如下:

- E 步骤:在当前的参数空间基础上计算 $Q(\Theta|\Theta^k)$
- M 步骤:更新参数空间,即用 Θ^{k+1} 代替 Θ ,要求使 $Q(\Theta|\Theta^k)$ 最大化

我们可以看出这两个步骤并不是一个详细的算法过程,还需要针对具体的问题进行细化,本书中介绍的很多算法实际上都是应用 EM 算法的实例,下面我们做一个简单的回顾,详细的算法过程参见相关章节。

Baum-Welch 算法 在 Baum-Welch 算法或者前向-后向算法(参见 9.3.3 节)中,E 步骤计算了两个值,一是在观察数据中每个状态 i 的转移数的期望值,二是每个状态对 (i, j) 从状态 i 到状态 j 的转移数的期望值。隐含数据相当于未观察到的状态转移。

M 步骤是在给定未观察数据后对参数的极大似然估计,在 Baum-Welch 算法中这些参数包

括初始的状态概率 π_i , 状态转移概率 a_{ij} , 以及符号发射概率 b_{ijk} 。

向内-向外算法 向内-向外算法(参见 11.3.4 节)中的未观察数据是指规则 $N^j \rightarrow \zeta$ 是否被用于生成某个特定的词序列 w_m , E 步骤计算就是该规则的期望使用次数。设定 $u_i(p, q, j, r, s)$ 代表生成终结节点的规则的期望使用次数, $v_i(p, q, j)$ 代表生成预终结节点的规则的期望使用次数(下标 i 是训练数据中的句子序号)。

M 步骤是基于 f_i 和 g_i 计算参数的极大似然估计; 向内-向外算法中的参数则是规则的概率。

无监督的词义消歧算法 7.4 节介绍的词义消歧算法非常类似于应用 EM 算法的高斯混合分布的参数估计问题, 仅仅是假设的概率分布模型不同。词义消歧算法是基于贝叶斯概率模型的, E 步骤计算隐藏二值变量 z_{ij} 的期望值(仅当词义 i 属于词义类别 j 时, $z_{ij} = 1$, 否则为零)。M 步骤则利用极大似然估计法计算聚类(聚类也可以解释为语义)生成某个特定单词的概率。

k 平均算法 k 平均算法可以看成是高斯混合分布聚类的一个特例。 k 平均算法实际相当于在 EM 算法的迭代过程中, 保持聚类的先验概率和协方差矩阵不变。在协方差很小的高斯分布模型的曲线图中, 可以观察到绝大多数的样本点聚集在中心位置, 因此如果样本属于第 j 个聚类的概率最大, 属于 j' 概率次之, 那么 $P(n_j | \hat{x}_i)$ 远远大于 $P(n_{j'} | \hat{x}_i)$ 。

这意味着在 E 步骤中计算后验概率时, 样本属于某个类别的概率接近于 1。但是, 在存在“等距”的情况下, 即样本可能和几个聚类类别的距离都相等时, k 平均算法是一种“硬”聚类方法, 样本只可能属于某一个聚类。而 EM 算法中允许一个样本同时属于几个聚类, 而且它们的概率值都相同。

总之, EM 算法是目前应用很广泛的一种算法, 但是不能忽视的是它本身也存在一些缺陷。首先, EM 算法对参数的初始值设置很敏感, “坏”的参数初始值设置甚至可以导致 EM 算法陷入一些局部最优点。为了避免发生这种情况, 我们可以先用其他的聚类算法给出初始参数的粗略估计值。例如, 在高斯混合分布的参数估计中, 利用 k 平均算法可以算出聚类中心点的有效初始估计值。另外, EM 算法的收敛速度比较慢, 而且 EM 的迭代过程只保证数据模型的改善, 对模型以外的其他许多事情并不具备改进效果。例如, 一个系统使用外部规则进行词性标注, EM 算法并不能提高其词性标注的准确率, 因为在这种情况下 EM 算法的优化对象和系统的评估函数之间并不匹配。最后, 很重要的一点是, 在解决问题时, 只有在不存在直接解决的算法的情况下, 我们才应该考虑使用 EM 算法, 因为它并不是一种解决限制条件下优化问题的高效方法。

14.3 深入阅读

读者如果希望阅读介绍性的入门级读物, 可以参见(Kaufman and Rousseeuw 1990)以及(Jain and Dubes 1988)。在信息检索系统中, 聚类方法也应用得比较广泛, 这方面的读物可以参阅(van Rijsbergen 1979; Rasmussen 1992; Willett 1988)等。

(Cormen et al. 1990: ch 24)论述了最小生成树算法, 设 n 为图中的节点数, 这类方法的算法复杂度一般为 $O(n \log n)$ 。

因其具备推广能力, 聚类算法通常用于数据分析和理解, 在一些数据可视化技巧中, 将高

维数据投影到低维空间(例如二维或者三维空间)上的过程是很关键的。目前有三种重要的方法,分别是主成分分析(principal component analysis, PCA)(Biber et al. 1998)、多维缩放法(Multi-Dimensional Scaling, MDS)(Kruskal 1964a, b)、自组织映射(Self-Organizing Map, 又称 Kohonen 映射)(Kohonen 1997),这些空间表示法都可以作为图 14.1 中树图的替代方法。

在一定程度上讲,聚类可以看成是感知器模型的一种类别规约形式,有些研究者曾经试图利用语料驱动的聚类方法进行语法类别规约(Brill et al. 1990; Finch 1993)。图 14.1 中的聚类方法是(Schütze, 1995)的方法,Waterman(1995)则希望借助于语法上的知识将词按照语义分类。

另外,一篇较早的论文(Hindle 1990)提出了和(Pereira et al. 1993)类似的方法,用于解决动词的宾语名词的聚类问题。Li and Abe (1998)建立了一个动词-宾语聚类模型,Basili et al. (1996)给出了动词的聚类算法,Hatzivassiloglou and McKeown(1993)讨论了形容词的聚类问题。

在大样本数据量条件下,聚类算法的效率显得相当重要。例如,Cutting et al. (1992)提出的 Buckshot 算法是一种高效聚类算法。关于另外一些高效的聚类算法,可以参阅(Cutting et al. 1993)和(Silverstein and Pedersen, 1997)。

14.4 习题

- 14.1 [★★]从网上下载一定数量的语料数据,选择其中的一个子集,分别采用单连通、全连通、平均连通聚类方法实现词聚类。
- 14.2 [★]构建一个样本集合,要求满足采用 k 平均算法聚类时迭代次数大于 1。
- 14.3 [★]在二维平面空间上,选择 10 个和原点距离小于等于 1 的样本点。在选择第 11 个点时,使其和原点的距离为(a) 2 ;(b) 4 ;(c) 8 ;(d) 16。针对每种情况,利用 k 平均聚类算法,分别采用两种空间的度量方式进行聚类;(1) 欧几里得空间;(2) L_1 空间(从靠近原点的 10 个点中选两个初始中心点)。结合聚类结果说明两种不同距离度量方式的不同点? 孤立点(第 11 个点)对聚类结果有什么影响?
- 14.4 [★★]EM 算法收敛于局部最优解,不同的初始参数会有不同的聚类结果。从网上选择 1000 个日常生活中最常用的词,应用 EM 算法分别选取 10 组不同的初始参数进行聚类。观察每次的结果之间有什么不同? 计算在所有 10 次聚类中,9 次聚类中,……,出现在同一类别中的词对的百分比。
- 14.5 [★]针对三种合并聚类算法之一或 k 平均算法,讨论如何均衡算法时间效率和聚类质量?
- 14.6 [★]Brown et al. (1992c)提出了一个 n 元语法模型,发现聚类能直接改进评测。但是在具体的实验中,发现只有经过线性差值运算才能改善评测,说明其可能的原因。
- 14.7 [★]假设样本数据中不存在“等距”(ties)情况,证明 k 平均算法的收敛性。同时将误差平方和作为聚类适合度的衡量标准,证明在算法迭代中这个值或者减少,或者保持不变。

第15章 信息检索

信息检索(Information Retrieval)用于通过特定算法或模型从文档中搜索有价值的信息,它是自然语言处理(NLP)的一个重要应用子领域,同时信息检索系统的特殊要求又使其区别于传统的自然语言处理,因为它覆盖了统计学领域的知识,以前这些领域在自然语言处理中并不是特别受关注。近年来自然语言处理中的“量化”方法再次成为研究的热点,随之而来是各个领域间相关性的日益增强。

本章选取4个主题进行介绍:文档词条(term)的概率分布模型(probabilistic models of term distribution)、向量空间模型(vector space model)、潜在语义索引(latent semantic indexing)和篇章分割(discourse segment)。其中,在潜在语义索引中我们引出特征空间降维的概念(dimensionality reduction)。本章的目的是希望读者能对信息检索有一个大致的了解,我们首先给出信息检索的一些基本概念,然后讨论这4个主题。

15.1 信息检索的背景

信息检索研究的目的是寻找从文档资料中获取可用信息的模型和算法,信息检索的传统问题是需要用户输入一个表述需求信息的查询字段,系统回复一个包含所需信息的文档列表,这一类问题称为特定信息的检索问题(ad-hoc retrieval problem)。为方便起见,我们在下面统称这类问题为 ad-hoc 问题,对于这类问题现在主要存在两种模型。精确匹配系统(exact match system)返回信息和检索信息之间的精确匹配,其中最有名的是布尔查询系统,目前这种方法仍然广泛应用于商业信息系统中。但是,对于大规模的异构文档集合,这类系统的查询结果往往为空集,所以近年来的研究转向按文档相关度返回查询结果的检索系统,我们将其作为本章的论述重点。这类系统的一个重要特点是和查询信息相关度大的文档优先返回,而且系统构建中涉及到了一些概率论的方法。

图 15.1 中给出了一个 ad-hoc 检索系统的例子,如图所示,用户想要查找罗浮宫的资料,所以他在 Alta Vista 搜索引擎中输入了查询字段“glass pyramid” Pei Louvre,搜索引擎返回了几个网站上的相关页面,但是我们可以看到,其中一些信息并不是用户期望得到的,即它们和用户所需要的资料毫不相关,相关性问题也正是信息检索系统有待解决的难点问题之一。

总的来说,ad-hoc 检索系统的研究涉及到如何通过相关度反馈(relevance feedback)来提高检索准确度,它要求用户对初始的查询字段进行优化;如何把从几个不同数据库中返回的结果合并到一个结果集当中;如何选择正确的模型,从在一定程度上损坏了的文档中检索信息,比如 OCR 识别的文档可以看成是此类文档;如何解决在非英文系统环境下的检索等一些难题。

在信息检索的一些子领域中,检索算法的建立依赖于训练语料库中文档的分类(与某类查询是否相关)。在文本分类问题(text categorization)中,用户希望能将文档划分为预先定义的两类或者更多类的文档类别。它的用途很广,例如报纸根据其报道信息,将其分为体育类、娱乐类等很多类别。路透社新闻文档的分类(Lewis, 1992)是将其分为 CORP-NEWS(corporate news),

CRUDE(crude oil), ACQ(acquisitions)等几类,从而方便订阅者的阅读,可以通过特定的应用程序只索取他们感兴趣的内容。

[AltaVista] [Advanced Query] [Simple Query] [Private eXtension Products] [Help with Query]

Search the **Web Usenet**
Display results **Compact Detailed**

Tip: When in doubt use lower-case. Check out Help for better matches.

Word count: glass pyramid: about 200; Pei:9453; Louvre:26578

Documents 1-10 of about 10000 matching the query, best matches first.

Paris, France

Paris, France. Practical Info.-A Brief Overview. Layout: One of the most densely populated cities in Europe, Paris is also one of the most accessible,...

<http://www.catatravel.com/paris.htm> - size 8K - 29 Sep 95

Culture

Culture. French culture is an integral part of France's image, as foreign tourists are the first to acknowledge by thronging to the Louvre and the Centre..

<http://www.france.diplomatie.fr/france/edu/culture.gb.html> - size 48K - 20 Jun 96

Travel World - Science Education Tour of Europe

Science Education Tour of Europe. B E M I D J I S T A T E U N I V E R S I T Y Science Education Tour of EUROPE July 19-August 1, 1995...

<http://www.omnitavel.com/007etour.html> - size 16K - 21 Jul 95

<http://www.omnitavel.com/etour.html> - size 16K - 15 May 95

FRANCE REAL ESTATE RENTAL

LOIRE VALLEY RENTAL. ANCIENT STONE HOME FOR RENT. Available to rent is a furnished, french country decorated, two bedroom, small stone home, built in the..

<http://frost2.flemingc.on.ca/~pbell/france.htm> size 10K - 21 Jun 96

LINKS

PAUL'S LINKS. Click here to view CNN interactive and WEBNEWSor CNET. Click here to make your own web site. Click here to manage your cash. Interested in...

<http://frost2.flemingc.on.ca/~pbell/links.htm> size 9K - 19 Jun 96

Digital Design Media, Chapter 9: Lines in Space

Construction planes... Glass-sheet models... Three-dimensional geometric transformations... Sweeping points... Space curves... Structuring wireframe...

<http://www.gsd.harvard.edu/~malcolm/DDM/DDM09.html> size 36K - 22 Jul 95

No Title

Boston Update 94: A VISION FOR BOSTON'S FUTURE. Ian Menzies. Senior Fellow, McCormack Institute. University of Massachusetts Boston. April 1994. Prepared..

<http://www.cs.umb.edu/~serl/mcCormack/Menzies.html> size 25K - 31 Jan 96

Paris - Photograph

The Arc de Triomphe du Carrousel neatly frames IM Pei's glass pyramid, Paris 1/6. © 1996 Richard Nebesky.

图 15.1 在网络搜索引擎中输入“glass pyramid”Pei Louvre 之后的搜索结果

过滤(filtering)和信息路由(routing)是文本分类的两个特例应用。之所以称为特例应用,是因为实际上它们都是二值分类问题。在 routing 系统中,系统输出结果是按照估计相关度排序的文档,与图 15.1 中的 ad-hoc 检索系统类似,但与 ad-hoc 系统的一个显著的不同点是 routing 系统中训练集文档的类别信息是已知的,其典型应用是邮件分发系统。在 filtering 系统中,需要计算的是每个文档的相关度,而把相关度小于某一个阈值的文档过滤掉,相关度用概率值的形式表示。filtering 系统的实现难度比 routing 系统大,因为它需要文档的绝对相关度(文档 1 是相关的)而非相对相关度(文档 1 比文档 2 相关度大)。在实际应用中,这是很必要的,因为如果用户在一个新闻组中查找一个主题,肯定希望立即得到相关的文档,而不是等到所有的相

关文档按相关度计算并排序之后得到一个列表。因为,对于用户来说,他不可能等上几天甚至一个月的时间,等到系统找到所有的相关新闻,然后再把其中相关度高的排在前面返回给用户,显然这种时间上的耗费是不可能被接受的。

15.1.1 信息检索模型的基本设计

绝大多数信息检索系统都为它们的主要数据建立倒排索引。倒排索引(inverted index)是这样一个数据结构,它列出了每个单词所在的所有的文档,以及在每一篇文档中出现的频次,建立倒排索引可以大大方便查询字段的查找,查询系统只需要在索引中找到该单词,就可以确认它的出现文档。

一个优化的倒排索引也包含单词的位置信息,除了上面提到的信息之外,在索引中还列出了单词在文档中的位置。这样,利用包含位置信息的索引就可以查找短语字段,其中每个单词的位置用距离文档起始位置的偏移值来表示,含有单词位置信息的倒排索引文件为搜索短语提供了可能。例如,用户输入查询字段 *car insurance*,我们可以在倒排文档中同时查找包含 *car* 和 *insurance* 的文档。首先将包含两个字段的文档集进行交集运算,得到它们同时出现的文档集合,然后通过单词的位置信息,确定保留那些 *car* 出现在 *insurance* 之前的文档,即需要的结果。这种方法显然要比处理所有文档的方法效率高得多。

这里提到的短语概念是相当狭义的,指固定的结构。例如,查找包含 *car insurance rates* 的文档时并不能找到包含 *rates for car insurance* 的文档,尽管它们的语义相同。这也是来自自然语言处理的一个有待进一步研究和解决的问题。近年来大多数信息检索系统都单独设计了一个短语的识别模块,然后为识别出的短语建立文档索引,在这种系统中短语处理和词的处理算法大致相同。最简单的短语识别方法是将那些出现频度最高的二元结构作为短语,例如将出现次数大于 25 的二元结构都作为短语,虽然简单,但效果却出奇得好。

我们可以看出,短语识别模块非常类似于寻找词语搭配的问题,所以很多词的搭配算法可以应用于此,相信在一定程度上可以提高短语索引和搜索的性能。

信息检索系统并不是为所有单词都建立索引文件。例如,一些语法词和功能词不需要建立索引。如表 15.1 所示,其中最常见的功能词包括单词 *the*, *from* 和 *could*。此类词的数量有限,可以把它们收集起来,组成一个词表,称为停用词表(stop list)。尽管这些功能词具备很强的语义功能,但它们对查询的贡献很小。

表 15.1 英文的停用词表,停用词表主要是一些功能词,这些功能词在信息检索中对于检索的准确率影响很小

a	also	an	and	as	at	be	but	by
can	could	do	for	from	go			
have	he	her	here	his	how			
i	if	in	into	it	its			
my	of	on	or	our	say	she		
that	the	their	there	therefore	they			
this	these	those	through	to	until			
we	what	when	where	which	while	who	with	would
you	your							

排除这些词还有一个好处,即大大减少了索引文件的大小。以英文为例,按照 Zipf 法则(见 1.4.3 节),排除停用词表后大概能减少 1/2 的索引文件规模。基于这种原因,实际上很多

检索系统都不为停用词表中包含的词建立索引。但是,一旦把停用词表中的单词排斥在检索系统之外,那么查找包含这些词的短语就不可能了,例如短语 *when and where*, 它的组成单词全部都是停用词表中的单词。

信息检索系统的另外一个常见问题是词干化(stemming),我们已经在 4.2.3 节对其进行了一些讨论。在信息检索系统中词干化分析意味着词的形态分析,通常的做法都是采用截断法,即去掉词缀取单词的一部分。词根通常是指一个单词的原形,例如单词 *laughing*, *laugh*, *laughs* 和 *laughed* 都是从 *laugh*- 演化而来的。词干化带来了两个问题,一方面语义不同的单词如 *gallery* 和 *gall* 往往具有相同的词干 *gall*-; 另一个问题是词干对于用户来说往往是不清楚的,如果我们查询 *gall*-, 如何处理 *gallery* 就成了一个问题。所以说对于形态变化很丰富的语言来说,这是一个比较难处理的问题。

15.1.2 评估准则

信息检索系统的性能取决于它们如何有效地将检索到的相关文档返回给用户,目前可应用的检验系统性能的衡量标准有很多,但是对于相关性文档的排序,信息检索系统中定义了专门的评估标准。它们大多数都涉及了准确率(precision)和召回率(recall)两个概念,准确率是指在查询返回的结果集中相关文档所占的百分比,召回率是指得到的相关文档占所有的相关文档集的比例。

我们用下面所示的表来说明为什么返回文档的排序是很重要的(见表 15.2)。在下面的例子中所有的三个检索结果集中都包含有相同数目的文档数。根据它们的准确率(都为 50%)无法比较三个结果的好坏,但是从直觉上观察,很明显结果 1 的效果要好于其他两个,所以这样一个简单定义的准确率并不能满足评估的要求。

表 15.2 文档排序评估的例子。表中列出了三个返回的查询结果,每个结果都包含 10 个文档,其中 R 表示该文档相关, N 表示不相关。我们基于四种不同的准则对排序进行了评估:截至位置 5 的准确率、截至位置 10 的准确率、非插值平均准确率、插值平均准确率(11 点)。具体的定义在下面给出

评估	排序 1	排序 2	排序 3
	D1: ✓	D10: x	D6: x
	D2: ✓	D9: x	D1: ✓
	D3: ✓	D8: x	D2: ✓
	D4: ✓	D7: x	D10: x
	D5: ✓	D6: x	D9: x
	D6: x	D1: ✓	D3: ✓
	D7: x	D2: ✓	D5: ✓
	D8: x	D3: ✓	D4: ✓
	D9: x	D4: ✓	D7: x
	D10: x	D5: ✓	D8: x
截至位置 5 的准确率	1.0	0.0	0.4
截至位置 10 的准确率	0.5	0.5	0.5
非插值平均准确率	1.0	0.3544	0.5726
插值平均准确率(11 点)	1.0	0.5	0.6440

一种度量方法是采用特定截取值的准确率,例如在上表中我们可以只求结果中前 5 个文档的准确率,通过观察这些值,获取相关文档和非相关文档排列的基本信息。

非插值平均准确率(uninterpolated average precision)的方法通过将几个不同的准确率结合到一个值中,来反映效果的好坏。如果我们把结果集的排列文档看成是按一定顺序排列的点,每一个点对应于一篇文档,这种方法就是计算那些相关文档点的准确率的均值。举例来说,在上表中我们在结果集 1 中的每一个相关文档点得到的准确率值都是 1,所以最后的结果也是 1。但是,对于结果集 3,在相关文档排序点上我们得到的准确率是 $1/2(d_1)$, $2/3(d_2)$, $3/6(d_3)$, $4/7(d_5)$ 和 $5/8(d_4)$, 所以最后的结果就是取这些值的均值 0.5726。

如果结果集之外不存在其他相关文档,那么这种方法就是假定这些漏掉的相关文档数为 0,所以我们可以说在一定程度上这个平均准确率反映了召回率,因为对于那些没有包含在结果集中的文档,我们都假定它们的准确率为 0。

我们进一步定义插值准确率均值(interpolated average precision),它更直观地反映了召回率大小,计算在一些特定召回率点上的准确率,并取它们的均值。例如,最常用的方法是取 11 点均值法,即在召回率为 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 100% 的文档排序的点上计算准确率。具体方法如下:对一个召回率 α , 在召回率为 α 的文档排序点上计算准确率 β , 但是如果在文档集中紧接着它的文档点的准确率出现上升,我们就需要沿着文档列表进行计算,一直到准确率下降为止。例如,对于图 15.2 中的排序 3, 在召回率为 60% 的点上,它的准确率并不是 $4/7$, 而是 $5/8(5/8 > 4/7)$ 。这样做的目的是基于这样一个思想,即如果结果的准确率一直在增加,那么用户当然希望在文档列表中看到更多的相关文档。在图 15.2 中我们就计算了 0%, 20%, 40%, 60% 和 80% 这 5 个点值(假设 5 篇文档就是相关文档的全集)。将准确率和召回率的关系画成图 15.2, 它更加清晰地表达了两者的关系,我们称之为 precision-recall 曲线。

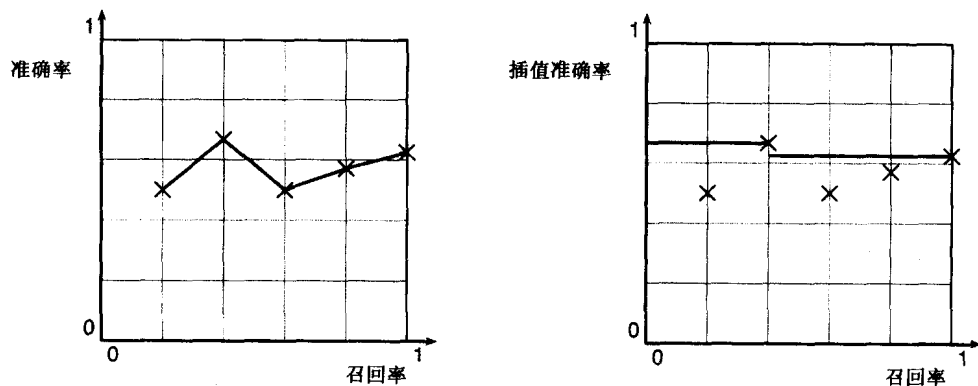


图 15.2 两条曲线分别表示表 15.2 中排序了非插值和插值的准确率-召回率关系曲线

很明显同时取得高的召回率和准确率是不可能的。得到高召回率的同时,准确率往往很低。例如,如果将相关文档全部召回,召回率为 100%,那么准确率就不可能很高,反之亦然。通常需要在召回率和准确度之间取得一个折中,前面所述的准确率均值是一个常用的方法,另外一个常用的方法称为 F 测量,我们已经在 8.1 节中做了相应的介绍。

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} \quad (15.1)$$

其中 P 是准确率, R 是召回率, α 是准确率和召回率权重因子。

所有这些度量都可以作为信息检索系统的评价准则,一般做法是首先在一定规模语料和

查询条件上运行信息检索系统,然后将各个查询的性能平均化后的结果作为评估标准。如果系统1的平均性能高于系统2,那么我们可以说系统1好于系统2。

但是这种试验方法的设计也存在不足之处。也就是说,有时候存在特殊情况,除了有一个查询在一个系统上比在另一个上的执行效果好之外,所有其他的查询字段测试的效果都相同,但这并不能说明系统之间的优劣与否,所以说均值法也存在着一定的偶然性,明智的做法是采用统计学上的方法,如 t 检验(参见 6.2.3 节)。

15.1.3 文档概率排序准则

从直观上来说,将得到的文档集合进行排序有利于用户查找。例如,如果用户得到的第一页结果的召回率很低,就是说找到用户所需信息的概率很小,那么系统在返回下一页的查询结果时,可以调整召回率和准确率的关系,适当地偏向召回率。

现在几乎所有的检索系统都基于文档概率排序准则(probability ranking principle)。为了便于理解,下面给出一种比较简明的阐述方式。

文档概率排序准则 (PRP):将文档按照其相关概率的降序排列是最优的文档排序方法。

其基本思想是把信息检索看成是确认“最有价值”文档的一个贪婪算法。所谓的“最有价值”是指文档的相关概率最大,即 $P(R|d)$ 的值最大。而一系列类似的判定结果就使我们得到了一个按照相关概率大小排列的文档序列,相关概率大的文档排在前面。

基本上现有的信息检索系统都是基于文档概率排序准则构建的,所以理解这个原则对于建立信息检索系统非常重要。这个准则本身隐含了两个基本假设前提,其中一个假设前提是文档集中文档之间是不相关的。但是,一个明显的反例就是文档副本。我们有两个相同的文档 d_1 和 d_2 ,假设 d_1 在前,但是 d_2 的出现并不能提供任何有别于 d_1 的信息,按常理它在返回的文档列表中的位置不应该很靠前,因为和它相同的文档 d_1 已经作为结果返回给查询用户了,当然最好的方法是只给出相同文档中的一个,但这显然和 PRP 的基本原则相违背。

另外一个假设前提是,复杂的初始查询字段可分解成单独的子查询,并且把它们各自优化,分别计算文档对每一个子查询的相关概率,归并每个子查询的结果就可以得出文档对于初始查询字段的相关概率。实际上,一个和整个查询字段相关的信息对中间步骤可能并不是最优结果,明显的反例是歧义字段的查询,例如用户输入歧义查询字段 *jaguar*,它既可能是动物名称(美洲虎),同时也可能是一种汽车牌子(英国产“美洲虎”牌汽车),而理想的检索系统应该对这种情况做出判断,必要的时候进行一定的消歧处理。但是,基于 PRP 准则建立的信息检索系统对这种歧义不做区分,也就是说,它返回的结果既有可能是动物,也有可能是一种汽车。

最后需要提醒读者的是,表示相关度的概率仅仅是一个估计值。而且,在对信息检索系统做了很多预先的假设后,得到的概率值是不是正确地反映了文档的相关程度,这一点本身就值得讨论。例如,需要考虑概率的方差,因为具有高相关概率的文档的概率方差值可能很大。

15.2 向量空间模型

向量空间模型是一种广泛应用于 ad-hoc 类信息检索的模型,主要原因是它具备概念简单,应用方便,以及利用空间相似性来逼近语义相似性等优点。文档和查询都在高维空间表示,其

中空间的每一维都表示文档中的一个单词。如果文档向量和查询向量距离最近,那么这个文档和查询相关程度最大,它们之间的距离通常用向量之间的夹角来表示。

如图 15.3 所示,我们设定向量空间为二维,对应于单词 *car* 和 *insurance*,查询字段的向量空间表示为 $(0.71, 0.71)$,三篇文档 d_1, d_2 和 d_3 的值分别对应为: $(0.13, 0.99)$, $(0.8, 0.6)$ 和 $(0.99, 0.13)$ 。稍后介绍这些值如何计算得来。我们称这个值为词条的权重(不称之为单词权重的原因在于空间模型中的元素既可能是单词也可能是短语)。

图中 d_2 和 q 的夹角最小,所以它和查询的相关程度最大。如果查询是 *car insurance*,在返回的相关文档中最前面的文档就是 d_2 ;因为 *car* 和 *insurance* 在 d_2 中很突出,因而有高的权重。

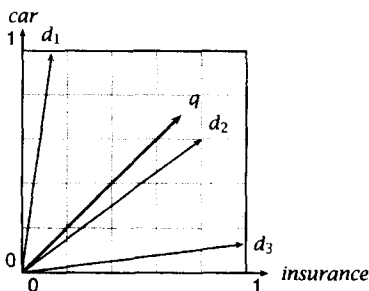


图 15.3 二维空间向量,一维表示单词 *car*,另一维表示单词 *insurance*

15.2.1 向量的相似度

在向量空间进行信息检索,需要计算向量夹角的余弦值或者归一化的相关系数(normalized correlation coefficient)(参见 8.5.1 节)

$$\cos(\vec{q}, \vec{d}) = \frac{\sum_{i=1}^n q_i d_i}{\sqrt{\sum_{i=1}^n q_i^2} \sqrt{\sum_{i=1}^n d_i^2}} \quad (15.2)$$

其中, $\vec{d}$ 与 $\vec{q}$ 都是 n 维实空间上的向量,计算这个值的大小可得到文档和查询的相关程度,公式中考虑到了文档长度的影响,即用它们的向量相关度除以两个向量范数的乘积。

回顾 8.5.1 节,对于归一化向量余弦和欧氏距离会给出相同的排序:

$$\begin{aligned} (|\vec{x} - \vec{y}|)^2 &= \sum_{i=1}^n (x_i - y_i)^2 \\ &= \sum_{i=1}^n x_i^2 - 2 \sum_{i=1}^n x_i y_i + \sum_{i=1}^n y_i^2 \\ &= 1 - 2 \sum_{i=1}^n x_i y_i + 1 \\ &= 2(1 - \sum_{i=1}^n x_i y_i) \end{aligned} \quad (15.3)$$

对于特定的查询向量 $\vec{q}$ 和两个文档 $\vec{d}_2$ 和 $\vec{d}_1$, 有以下的公式,表明余弦法和利用向量欧几里得距离法得出的结论一致:

$$\cos(\vec{q}, \vec{d}_1) > \cos(\vec{q}, \vec{d}_2) \Leftrightarrow |\vec{q} - \vec{d}_1| < |\vec{q} - \vec{d}_2| \quad (15.4)$$

向量空间模型应用的潜在前提是文档长度和相关度无必然联系,假设事先没有对向量的相关系数进行归一化处理,显然长篇幅文档的相关度要比短的高,因为它包含更多单词(我们将其留为练习,以表明图 15.3 中的向量是归一化的,即 $\sqrt{\sum_i d_i^2} = 1$)。

15.2.2 词条权重

本节讨论如何计算向量空间中词条的权重,最简单的计算方法是将词条在文档中的出现频率作为权重,但是在实际运用中往往采用其他更有效的计算方法。计算词条权重的一个重要信息是词条频度 (term frequency)、文档频度 (document frequency) 和收集频度 (collection frequency),定义如表 15.3 所示。

表 15.3 信息检索中计算词条权重常用的三种度量

度量	符号标识	定义
词条频度	$tf_{i,j}$	单词 w_i 在 d_j 文档中的出现次数
文档频度	df_i	出现单词 w_i 的文档数
收集频度	cf_i	单词 w_i 出现的总次数

在表 15.3 中, $df_i \leq cf_i$, $\sum_j tf_{i,j} = cf_i$, 文档频度和收集频度的计算都是以文档集合的存在为前提的。如果文档集合总是在不断变化,比如文档集合是随机通过从几个大型数据库中抽取数据形成的临时集合,这个前提就不成立了。

词条频度描述了词条在给定文档中的重要程度,这个值越大,表示该词对文档的描述程度越高,更准确地反映了文章的内容。通常,需要对这个值取开方或者取对数做平滑,即 $f(tf) = \sqrt{tf}$ 或 $f(tf) = 1 + \log(tf)$, $tf > 0$, 平滑后的数值更能反映它的内在含义。虽然词频越高,越能体现词和文档内容的相似性,但是在一篇文档中如果某个词的出现次数是在另外一个文档中出现次数的 3 倍,并不能表明相关性也具备倍数关系。

文档频度表明词条的信息度 (informativeness)。如果一个语义中心词只局限在一篇文档中,那么它在该文档中的出现次数可能很多;与此相反,一个非语义中心词在文档集合的所有文档中的分布是相对均匀的。也就是说,如果该词在每个文档中都可能出现,很显然这说明这个词和文档内容的相关度不大。这里有个例子,是单词 *insurance* 和 *try* 在纽约时报语料库中的统计信息,统计数字表明两个词的出现频率相差无几(见表 15.4)。

表 15.4 实例语料库中两个词的词条和文档频度

单词	收集频度	文档频度
<i>Insurance</i>	10 440	3 997
<i>Try</i>	10 422	8 760

但是,出现 *insurance* 的文档大概只有 *try* 的一半。究其原因,主要是因为单词 *try* 在任何文档中都有可能出现,但是单词 *insurance* 表达的概念要狭隘得多,大概平均 1 篇文档出现 3 次,局限于医疗保险、汽车保险等内容的文档。

当然最理想的情况是同时把词条频度 $tf_{i,j}$ 和文档频度 df_i 的信息结合在一个公式中,方法如下:

$$\text{weight}(i, j) = \begin{cases} (1 + \log(\text{tf}_{i,j})) \log \frac{N}{\text{df}_i} & \text{若 } \text{tf}_{i,j} \geq 1 \\ 0 & \text{若 } \text{tf}_{i,j} = 0 \end{cases} \quad (15.5)$$

其中 N 为文档集中的总文档数,使用第一个公式的前提是词频大于 0, $\log\left(\frac{N}{\text{df}_i}\right)$ 在公式中很关键,当 df_i 为 1 时,这个值最大,当 df_i 为 N 时,这个值最小(为 0),即词出现的文档数越多,整个权重的值就越小,这也和前面的讨论结果相符。

这种形式的文档频度权重值称为倒排文档频度(inverse document frequency)或 idf 权重。上面的公式仅仅是 $\text{tf} \cdot \text{idf}$ 权重计算公式族的一个特例而已, $\text{tf} \cdot \text{idf}$ 权重计算公式族中的每个公式都包含了词条频度($\text{tf}_{i,d}$)、文档频度(df_i)和归一化信息(式(15.5)中无)。我们可以把这种不同组合的方案列成表的形式,每一种组合就是一种权重的计算方法,如表 15.5 所示。

表 15.5 $\text{tf} \cdot \text{idf}$ 权重计算公式族, 其中 $\text{tf}_{i,d}$ 为词条 t 在文档 d 中的出现次数, df_i 为出现词条 t 的文档数目, N 为文档总数

词条频度的计算		文档频度的计算		归一化的方法	
n(natural)	$\text{tf}_{i,d}$	n(none)	df_i	n	(no normalization)
l(logarithm)	$1 + \log(\text{tf}_{i,d})$	t	$\log \frac{N}{\text{df}_i}$	C	(cosine)
a(augmented)	$0.5 + \frac{0.5 \times \text{tf}_{i,d}}{\max_d(\text{tf}_{i,d})}$				$\frac{1}{\sqrt{w_1^2 + w_2^2 + \dots + w_n^2}}$

依照定义,可把公式(15.5)简称为 ln, l 代表对词频取对数加 1, t 代表对文档频度的倒数乘 v 再取对数, n 表示公式没有归一化处理。但是这些都是经验公式,没有经过严格数学论证,但在实际应用中它们都是相当有效的,所以在一些准确度要求不是很严格的情况下可以应用这些公式。

15.3 词条分布模型

计算词条权重的另一个很自然的想法就是构建一个词条分布的概率模型,并利用这个模型来衡量词条对于检索的重要性。也就是说,我们希望估计单词 w_i 在文本中出现 k 次的概率 $P_i(k)$ 。在最简单的情况下,分布模型可以用来为向量空间模型推导出一个概率驱动的词条权重分配方法。同时,也可以把它嵌入到信息检索框架中。

词条分布模型除了在词条权重设置中 useful 以外,它对于文本中词的出现模式的精确描述至少也与统计自然语言处理中的另一个重要话题(Zipf 法则)同等重要。Zipf 法则描述了整个语料库中词条的行为,即出现模式。与之相比,词条分布模型抓住了语料库子单元(文本或书的章节)中词条出现的规律。除了信息检索,知道词条分布模式对于评估某个特定词条在文本中出现特定次数的可能性也很有用。例如,在评估一个未知版权的作品到底属于哪个作家时就很有用。

大多数词条分布模型都致力于评价词条所包含的信息量,这也正是倒排表所做的。这个问题可以简化为区分实词和虚词,但是大多数模型都有自己的词条权重等级。在这一节中,我们介绍了几种不同的模型来描述信息量的含义。其中有三个模型基于泊松分布,一个把倒排

表作为贝叶斯分类的最理想的权重分配方法,最后一个剩余倒排表,它可以简单解释为倒排表与泊松分布的结合。

15.3.1 泊松分布

标准泊松分布是描述特定事件在一个固定大小的空间(一段时间,或者是一定体积的液体)中的分布情况。例如一段时间内的次品率,以及一页上的错误字符数都是典型的泊松分布例子,泊松分布的定义如下:

$$\text{泊松分布: } p(k; \lambda_i) = e^{-\lambda_i} \frac{\lambda_i^k}{k!} \quad \text{对于某些 } \lambda_i > 0$$

由概率论可知,在泊松分布中样本的期望值和方差都是 λ_i 。具体到信息检索领域中, λ_i 是单词 w_i 在一篇文档中的平均出现次数,等于 $\lambda_i = \frac{cf_i}{N}$, cf_i 是文档频度, N 是文档集合的总文档数。

图 15.4 是两个泊松分布的例子。

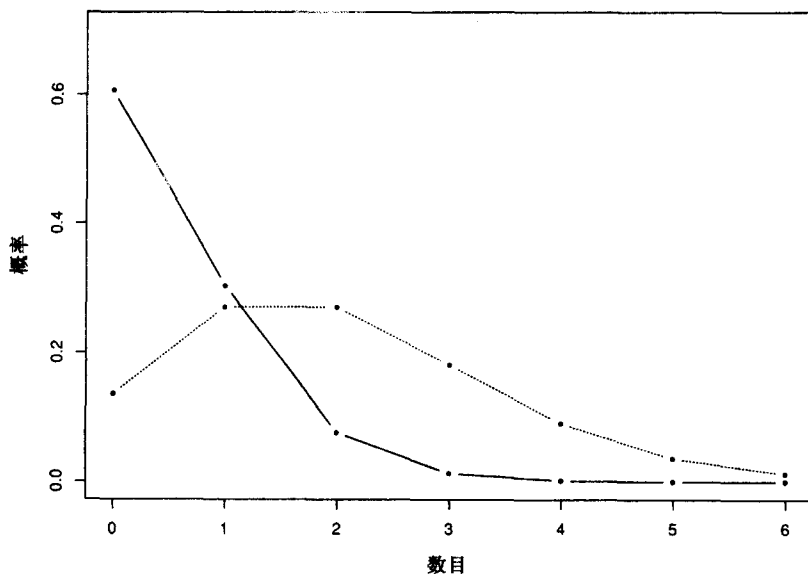


图 15.4 泊松分布,图表中的实线表示 $p(k, 0.5)$,虚线表示 $p(k, 2.0)$, k 表示词的出现次数 ($0 \leq k \leq 6$)

信息检索系统所感兴趣的是一个特定词条在固定大小的单位(例如一篇文档)中的分布情况,那么一个单词在一篇文档中出现一定次数的概率值等于什么呢?按照上面的定义,我们可以用 $P_{i(k)} = p(k; \lambda_i)$ 来表示单词 w_i 在文档中出现 k 次的概率,其中每个单词 λ_i 的值都需要单独统计计算。

根据概率的知识,二项分布在极限情况下可以用来代替泊松分布。对于二项分布,如果 n 趋向于无限大, p 值趋向于 0,那么二项分布 $b(x; n, p) \rightarrow p(k; \lambda)$ 。用泊松分布来近似词条分布需要满足以下 3 个条件:

- 在短篇幅文档中词条出现一次的概率和该篇文档的长度成正比。
- 词条出现一次以上的概率远远小于词条出现一次的概率。

- 事件的出现文档各部分相互独立。

表 15.6 中展示了对于纽约时报中的 6 个词条,泊松分布是怎样预测其文档频率的。对于每个词,文档频率为 df_i ,收集频率为 cf_i ,估计 λ (收集频率除以文档总数 79 291),预测 df ,预测 df 与实际 df 的比率。

检测文档频率是检测一个词是否是泊松分布的最简单的办法。某个词出现至少一次的文本数目可以通过对不出现的该词的文本数目求补得到。因此,泊松分布估计出来的文本频率为 $\widehat{df}_i = N(1 - p(0; \lambda_i))$, N 是语料库中文本的数目。检查泊松分布的一个比较好的办法是查看完全分布,即检查词语出现 0, 1, 2, 3, ... 次的文本数目。做法如下。

在表 15.6 中,我们可以看到泊松估计对于类似于 *follows* 和 *transformed* 之类的虚词效果是很好。我们用“虚词”来描述那些对文本内容没有什么贡献的词。但是,对实词的估计是很高的,大约为 3(3.48 和 2.91)。

表 15.6 统计文档频率,收集频率实际值,通过泊松公式计算估计值及其误差。当根据虚词 (*follows*) 计算文档频率时公式误差很小,但是由实词 (*soviet*) 得出的文档频率则和真实值相差很大。语料库中共有 79 291 个文档

单词	df_i	cf_i	λ_i	$N(1 - p(0; \lambda_i))$	估计的误差
<i>Follows</i>	21 744	23 533	0.2968	20 363	0.94
<i>Transformed</i>	807	840	0.0106	835	1.03
<i>Soviet</i>	8204	35 337	0.4457	28 515	3.48
<i>Students</i>	4953	15 925	0.2008	14 425	2.91
<i>James</i>	9191	11 175	0.1409	10 421	1.13
<i>Freshly</i>	395	611	0.0077	609	1.54

如果考虑到假设的独立性前提,结果其实并不令人意外,因为实词往往不是独立出现的。单词出现一次以后,它再次出现的概率就会很大,此种性质也称为实词的聚类性 (term clustering),所以可使用单词的非泊松分布特性来判断其是否具备聚类特性。但同时并不能排除例外情况的出现,例如上表中 *james* 的出现,尽管它是实词,但是泊松分布却很好地拟合了它的分布。为什么会出现这种情况呢?最有可能的原因在于:如果一个人的全名出现了一次,那么后面可能出现其全名的可能性不会很大。按照日常用词的习惯,在下文提到这个人的时候通常以代词或者名字代替。另外,虚词 *freshly* 的分布并不和泊松分布相符,这也是出乎我们意料的。经过观察分析,出现这种情况的原因主要和纽约时报本身的语料有关,例如烹饪专栏的文章中经常会出现 *freshly* 这类词。

实词的聚类性给应用泊松分布带来了一些问题。但是事情总有两面性,小时候我们学习写作文的时候总被教导要避免重复用词,所以词在出现一次之后紧接着再次出现的概率总的来说不是很大。最后一个问题是文档大小对泊松分布的影响,因为文档大小不能作为一个衡量的尺度,如同“秒”对于时间,“千克”对于质量,但是应用泊松分布的一个前提就是要求忽略这样的影响。

15.3.2 二重泊松分布

一个更好的拟合实词分布的模型是二重泊松分布 (Bookstein and Swanson, 1975),它在本质

上是两个泊松分布的混合。对于每一个词条,这种模型将文档分为两类,第一类文档对应于该词条出现频度较低(non-privileged class)的文档,第二类对应于出现频度较高(the privileged class)的文档,模型的形式化定义如下:

$$tp(k; \pi, \lambda_1, \lambda_2) = \pi e^{-\lambda_1} \frac{\lambda_1^k}{k!} + (1 - \pi) e^{-\lambda_2} \frac{\lambda_2^k}{k!}$$

其中 π 是文档在第二类文档中的出现概率, $(1 - \pi)$ 对应于文档在第一类文档中出现的概率, λ_1 和 λ_2 为词条在第一类和第二类中出现次数的平均值 w_i 。

二重泊松分布假定实词在文档中扮演两种角色,在第一类文档中它等同于虚词,对信息检索其实并不起作用,不能作为索引词。而在第二类文档中,它作为中心词,对于检索的结果有很大的影响,可依照它的平均出现次数建立索引词条。实验表明这个结果对于频度等于2的单词分布的描述效果要差于频度等于3或者4的情况,进一步的研究发现,实际上 $P_i(k)$ 存在这样的关系, $P_i(0) > P_i(1) > P_i(2) > \dots$ 。作为改进方法,我们可用多重的模型来修正它。负二项分布(negative binomial)可描述无穷多个泊松分布的混合分布(Mosteller and Wallace 1984),当然负二项分布可以比一重或者二重泊松分布更好地拟合词条的分布,主要应用难点是分布参数的计算量巨大。

15.3.3 k 重混合分布

另外一个描述词条分布效果的比较好的模型是Katz提出的 k 重混合模型:

$$P_i(k) = (1 - \alpha) \delta_{k,0} + \frac{\alpha}{\beta + 1} \left(\frac{\beta}{\beta + 1} \right)^k$$

$$\text{其中, } \delta_{k,0} = \begin{cases} 1 & k=0 \\ 0 & \text{其他} \end{cases}$$

其他的参数计算如下:

$$\begin{aligned} \lambda &= \frac{cf}{N} \\ IDF &= \log_2 \frac{N}{df} \\ \beta &= \lambda \times 2^{IDF} - 1 = \frac{cf - df}{df} \\ \alpha &= \frac{\lambda}{\beta} \end{aligned}$$

参数 β 是每个文档中出数次数大于1的词条(特词条)数的均值, $\frac{\beta}{\beta+1} = \frac{cf - df}{cf}$ 实际上决定了

$\frac{P_i(k)}{P_i(k+1)}$ 的大小。例如,如果这个比值为0.1,则表示出现该词条一次的文档数是出现两次的文档数的10倍($P_i(k)$ 表示 w_i 在一篇文档中出现 k 次的概率)。这个结果不难理解。以此类推,出现这个词条两次的文档数是出现三次的文档数的10倍。同时也表示特词条的出现数是词条出现数的1/10。而如果 $\frac{\beta}{\beta+1}$ 等于0,则说明 w_i 在任何一篇文档中的出现次数都不大于1。

值得注意的是,模型可以准确地估计出词条出现次数为0的文档数,这也是模型非常好的一个特性。参数 α 表示词条的绝对出现频度信息。如果两词条有相同的 β ,表示它们有相同的收集频度和文档频度之间的比率。但是,如果收集频度不同,则参数 α 也不同。

表15.7中是纽约时报语料库统计的结果,从中可以看出,当 k 取0时,模型的描述效果较好,特别是对于虚词。但是,实词试验的结果不甚理想,和假设的 $\frac{P_i(k)}{P_i(k+1)} = c, k \geq 1$ 之间的差值很大。例如,以 *soviet* 为例,有如下的结果:

$$\begin{aligned} \frac{P_i(0)}{P_i(1)} &= \frac{71\,092}{3038} \approx 23.4 & \frac{P_i(1)}{P_i(2)} &= \frac{3038}{1277} \approx 2.38 \\ \frac{P_i(2)}{P_i(3)} &= \frac{1277}{784} \approx 1.63 & \frac{P_i(3)}{P_i(4)} &= \frac{784}{544} \approx 1.44 \\ \frac{P_i(4)}{P_i(5)} &= \frac{544}{400} \approx 1.36 \end{aligned}$$

表 15.7 6个单词出现 k 次的真实文档数和估计文档数,例如,单词 *follows* 出现两次的实际文档数目为1435,估计值为1527.3

Word		k									
		0	1	2	3	4	5	6	7	8	≥ 9
follows	act.	57 552.0	20 142.0	1435.0	148.0	18.0	1.0				
	est.	57 552.0	20 091.0	1527.3	116.1	8.8	0.7	0.1	0.0	0.0	0.0
trans-formed	act.	78 489.0	776.0	29.0	2.0						
	est.	78 489.0	775.3	30.5	1.2	0.0	0.0	0.0	0.0	0.0	0.0
soviet	act.	71 092.0	3038.0	1277.0	784.0	544.0	400.0	356.0	302.0	255.0	1248.0
	est.	71 092.0	1904.7	1462.5	1122.9	862.2	662.1	508.3	390.3	299.7	230.1
students	act.	74 343.0	2523.0	761.0	413.0	265.0	178.0	143.0	112.0	96.0	462.0
	est.	74 343.0	1540.5	1061.4	731.3	503.8	347.1	239.2	164.8	113.5	78.2
james	act.	70 105.0	7953.0	922.0	183.0	52.0	24.0	19.0	9.0	7.0	22.0
	est.	70 105.0	7559.2	1342.1	238.3	42.3	7.5	1.3	0.2	0.0	0.0
freshly	act.	78 901.0	267.0	66.0	47.0	8.0	4.0	2.0	1.0		
	est.	78 901.0	255.4	90.3	31.9	11.3	4.0	1.4	0.5	0.2	0.1

换句话说,文本中一个实词的每一次出现都减少了附加词条出现的概率,但这种减小的幅度在逐步变小。出现这种情况可能是由于实词的聚集性,特别是当文章的主题和实词语义相关时。而这种概念是这个模型所不能表达的。但是,总的来说,混合模型的效果显然要比上面提到的两种模型(泊松分布和二重泊松分布)的效果更好。

15.3.4 倒排文档频度

上节中引出了 IDF(inverse document frequency)的概念,本节中主要是从概率模型角度出发来推导这个概念。

为了引出这个概念,我们按照让步比(odds of relevance)排序 ad-hoc 信息搜索的文档。

$$O(d) = \frac{P(R|d)}{P(\neg R|d)}$$

其中 $P(R|d)$ 是文档 d 相关程度的概率, $P(\neg R|d)$ 是不相关的概率。为了计算简便,将其对数化,并应用贝叶斯公式简化,得到:

$$\log O(d) = \log \frac{P(R|d)}{P(\neg R|d)}$$

$$\begin{aligned}
&= \log \frac{\frac{P(d|R)P(R)}{P(d)}}{\frac{P(d|\neg R)P(\neg R)}{P(d)}} \\
&= \log P(d|R) - \log P(d|\neg R) + \log P(R) - \log P(\neg R)
\end{aligned}$$

现在假定查询字段 Q 是一个由 $\{w_i\}$ 组成的集合, 并设随机变量 $X_i = 0$ 或 1 , 来表示 w_i 在文档 d 中出现(1)还是不出现(0)。假设文档中词的分布独立, 那么可推出如下的结果:

$$\log O(d) = \sum_i [\log P(X_i|R) - \log P(X_i|\neg R)] + \log P(R) - \log P(\neg R)$$

因为我们感兴趣的仅是文档排序, 即排序函数 $g(d)$ 的相对值, 所以暂不考虑常数项 $\log P(R) - \log P(\neg R)$, 并简化表示为 $p_i = P(X_i = 1|R)$ (词 w_i 出现在相关文档中的概率), $q_i = P(X_i = 1|\neg R)$ (词 w_i 出现在非相关文档中的概率), 得到(在第二行的推导中使用了 $P(X_i = 1|_) = y = y^1(1-y)^0 = y^{X_i}(1-y)^{1-X_i}$ 和 $P(X_i = 0|_) = 1-y = y^0(1-y)^1 = y^{X_i}(1-y)^{1-X_i}$):

$$\begin{aligned}
g(d) &= \sum_i [\log P(X_i|R) - \log P(X_i|\neg R)] \\
&= \sum_i [\log(p_i^{X_i}(1-p_i)^{1-X_i}) - \log(q_i^{X_i}(1-q_i)^{1-X_i})] \\
&= \sum_i X_i \log \frac{p_i(1-q_i)}{(1-p_i)q_i} + \sum_i \log \frac{1-p_i}{1-q_i} \\
&= \sum_i X_i \log \frac{p_i}{1-p_i} + \sum_i X_i \log \frac{1-q_i}{q_i} + \sum_i \log \frac{1-p_i}{1-q_i}
\end{aligned}$$

其中 $\sum_i \log \frac{1-p_i}{1-q_i}$ 是常量, 不影响排序函数, 上式进一步简化为:

$$g'(d) = \sum_i X_i \log \frac{p_i}{1-p_i} + \sum_i X_i \log \frac{1-q_i}{q_i} \quad (15.6)$$

如果有一个按与查询相关与否分好类的文档集合, 从中可以直接得出概率 p_i 和 q_i 的大小, 但是在信息检索系统中这个信息通常是未知的。

这意味着必须定义一些必要的简化假设, 首先假设 p_i 的值对集合中所有的词条是一个小的常数, 所以 g' 的第一项变为 $\sum_i X_i \log \frac{p_i}{1-p_i} = c \sum_i X_i$, 即查询和文档的匹配数乘以系数 c 。而上式的第二项由如下假设来简化: 假设文档集合中的绝大多数文档和查询是不相关的, 相当于 $q_i = P(X_i = 1|\neg R) \approx P(w_i) = \frac{df_i}{N}$, 即对 $P(w_i)$ 的极大似然估计。这样 w_i 在非相关文档中出现的概率等于:

$$\frac{1-q_i}{q_i} = \frac{1-\frac{df_i}{N}}{\frac{df_i}{N}} = \frac{N-df_i}{df_i} \approx \frac{N}{df_i}$$

(因为相对来说在整个文档集合中每个词出现的文档数是很少的)。

经过两步的简化, 实际上推导出了和前面的结果(IDF)相似的结论, 即:

$$g'(d) \approx c \sum_i X_i + \sum_i X_i \text{idf}_i \quad (15.7)$$

注意,这个结果是根据 w_i 在非相关文档中而不是相关文档中出现的概率导出的,我们采用了一个迂回的计算方法,因为不可能直接利用在相关文档中出现的概率,这个信息是不可知的。

15.3.5 剩余倒排文档频度

此外还有一个倒排文档频度(IDF)的替代方法,称为剩余倒排文档频度(residual inverse document frequency),它定义为 IDF 与泊松模型估计之间的差值:

$$\text{RIDF}_i = \text{IDF}_i - \log_2 \frac{1}{1 - p(0; \lambda_i)} = \text{IDF} + \log_2(1 - p(0; \lambda_i))$$

其中 $\text{IDF}_i = \log_2 \frac{N}{\text{df}_i}$, p 为泊松分布概率密度函数, $\lambda_i = \frac{\text{cf}_i}{N}$ 是 w_i 的平均出现次数, $1 - p(0; \lambda_i)$ 表示按照泊松分布模型估计 w_i 至少出现一次的文档数。表 15.4 中的单词 *insurance* 和 *try* 的 RIDF 值分别为 1.29 和 0.16。 $N = 79\,291$, 试证明这一点!

但是,从上面章节的叙述中我们知道泊松分布仅仅适合于虚词分布,不过这也有一个好处,即它可以作为判断虚词的一个标准。

15.3.6 词条分布模型的应用

我们已经对词条概率分布模型做了简要介绍,并将其应用于衡量词条的相关度。例如 RIDF 或者 k 重混合模型中的 β 值,都可以作为词条权重。不言而喻,“好”的概率模型有利于更准确地对词条性质进行描述,信息搜索得到的结果也更合理。

15.4 潜在语义索引

以上介绍的所有方法关注的都是单个词条的出现,但是在信息检索中词条的同现也是重要的信息。这基于一个很简单的道理:文档中两个或者更多词条的出现不是偶然的事件。设想如下例子,在表 15.8 中,可能文档 1 对于查询字段而言具有更好的相关性,因为其中几乎包括了所有的查询词条。但是文档 2 也不失为一个好的选择,因为它表达的是一种语义相关的信息。潜在语义索引(Latent Semantic Indexing, 简称 LSI)就是一种探查这种内在语义联系的方法(由 *HCI* 和 *interaction* 与 *user* 和 *interface* 的同现关系得到)。潜在语义索引就是把同现词条映射到同一维空间上,而非同现词条则被映射到不同的空间上。从这个意义来讲,即使查询字段和文档没有共同词条,它们的余弦相似度可能仍然很高,其中的缘故就是同现分析。同现词条被认为是语义相关的。

潜在语义空间与我们原来的空间(维数和词条数相同)相比,其维数要小得多,因为很多词条被投影到同一维上,因此潜在语义索引(LSI)的方法其实是一种降维方法。维数消减过程就是将高维空间中的对象映射到较低维的空间上的过程,大多数情况下是二维或者三维的空间。图 15.5 是个很好的例证,这个矩阵定义在五维空间上(即词条 *astronaut*, *cosmonaut*, *moon*, *car*, *truck* 代表了五维),对象即是文档,用五个词条来表示。图 15.6 表示的是经过奇异值分解后的结果,直观地表现出了文档之间的关系,显然文档 d_4 和 d_5 , 文档 d_2 和 d_3 之间的相似度很高。而这些关系在图 15.5 中很难体现出来,其中 d_2 和 d_3 并没有共同的词条,但是潜在语义索引方法找出了它们的内在关系。

表 15.8 内容相似度计算中词条同现的实例,表中列出了查询和文档中包含的词条

	词条 1	词条 2	词条 3	词条 4
查询字段	User1	Interface		
文档 1	User2	Interface	HCI	Interaction
文档 2			HCI	Interaction

$$A = \begin{pmatrix} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \text{cosmonaut} & 1 & 0 & 1 & 0 & 0 & 0 \\ \text{astronaut} & 0 & 1 & 0 & 0 & 0 & 0 \\ \text{moon} & 1 & 1 & 0 & 0 & 0 & 0 \\ \text{car} & 1 & 0 & 0 & 1 & 1 & 0 \\ \text{truck} & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

图 15.5 词条-文档相关矩阵的实例 A

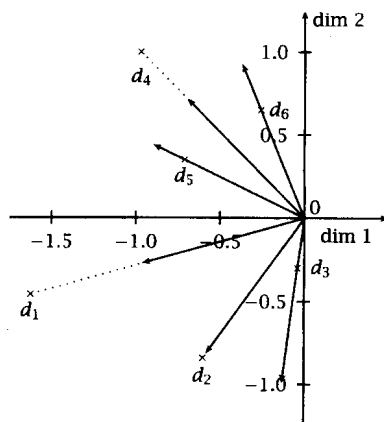


图 15.6 降维处理。图 15.5 中的矩阵的降维,从五维消减到二维,降维后的文档向量参见图 15.11

从空间降维角度讲,潜在语义索引(LSI)方法具备了某种最优特性,即它把原对象投影到方差最大的轴上,即数据在我们选取的映射轴上的投影具备最大的可分性。从这个意义上讲,它非常近似于平时所说的“主成分分析”(Principal Component Analysis),这是另一种降维法,不过 PCA 的方法仅仅适用于方阵,而潜在语义索引的操作对象可以是任何矩阵。

潜在语义索引主要利用了数学上奇异矩阵的分解方法(Singular Value Decomposition),它是一种满足最小方差原则的方法,即映射前后能保持原向量和投影向量之间的方差最小。

15.4.1 最小平方和法

在给出应用于 LSI 中的最小平方和的定义之前,首先介绍最小平方和的近似方法,即利用线性回归的方法拟合平面上的一组点,也就是我们通常所说的最小二乘法。

首先假设有一组点 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$,我们的目标是确定一条直线:

$$f(x) = mx + b$$

即确定参数 m 和 b 的值,使该直线和这些点的拟合度最大,在最小平方和意义下,意味着平方和最小。

$$SS(m, b) = \sum_{i=1}^n (y_i - f(x_i))^2 = \sum_{i=1}^n (y_i - mx_i - b)^2 \quad (15.8)$$

利用一阶导数的方法求极值

$$\begin{aligned}\frac{\partial SS(m, b)}{\partial b} &= \sum_{i=1}^n [2(y_i - mx_i - b)(-1)] = 0 \\ \Leftrightarrow [\sum_{i=1}^n y_i] - [m \sum_{i=1}^n x_i] - [nb] &= 0 \\ \Leftrightarrow b &= \bar{y} - m\bar{x}\end{aligned}\quad (15.9)$$

其中, $\bar{y} = \frac{\sum_{i=1}^n y_i}{n}$, $\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$, 它们是坐标 x, y 的均值。

我们将式(15.9)代入式(15.8)的 b 中, 并求解 $\frac{\partial SS(m, b)}{\partial m} = 0$, 得到:

$$\begin{aligned}\frac{\partial SS(m, b)}{\partial m} &= \frac{\partial \sum_{i=1}^n (y_i - mx_i - \bar{y} + m\bar{x})^2}{\partial m} = 0 \\ \Leftrightarrow \sum_{i=1}^n 2(y_i - mx_i - \bar{y} + m\bar{x})(-x_i + \bar{x}) &= 0 \\ \Leftrightarrow -\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) + m \sum_{i=1}^n (x_i - \bar{x})^2 &= 0 \\ \Leftrightarrow m &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}\end{aligned}\quad (15.10)$$

利用这些公式, 可计算出对应于点(1,1), (2,2), (6,1.5), (7,3.5)的结果如下(图 15.7 给出了相应的图示):

$$\begin{aligned}m &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{6.5}{26} = 0.25 \\ b &= \bar{y} - m\bar{x} = 2 - 0.25 \times 4 = 1\end{aligned}$$

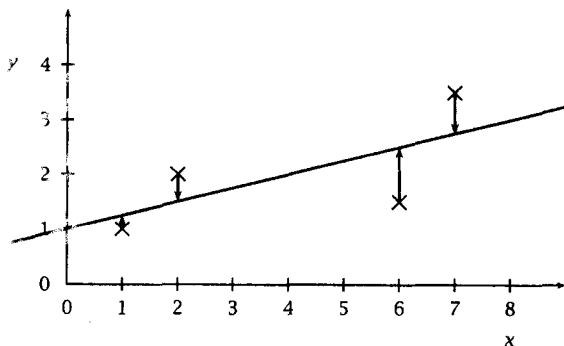


图 15.7 线性回归的例子, 图中的直线 $y = 0.25x + 1$ 是四个点 (1,1), (2,2), (6,1.5), (7,3.5) 的最小平方和估计

15.4.2 奇异值分解

正如我们所述, 利用潜在语义索引方法可以分析文档中词条的同现, 而不是简单地计算词条的重叠, 如余弦法那样。奇异值分解方法能起到降维作用, 降维的同时, 同现词条被投影到降维后空间的同一维上, 从而提高了具备相同语义的文档之间的相似度计算值。词条同现分析和降维是理解潜在语义索引的关键, 所谓潜在语义索引实际上是词条-文档关系矩阵的奇异值分解方法(SVD)在信息检索中的一个应用。SVD 就是将矩阵 A 表示为降维处理后的矩阵 $\hat{A}$, 并使下面的标准二次型的值最小化, 奇异值分解也是计算向量之间的欧几里得距离。图 15.7 可

以解释为将一些二维的点投影到直线上降为一维的点,与此相对应的是,SVD的方法是将一个 n 维空间映射成 k 维($k \ll m$),把文档向量空间模型从 m 维转变到维数较低的空间上。

$$\Delta = \|A - \hat{A}\|_2 \quad (15.11)$$

可能会有人对上式存在一些困惑,不是矩阵 $\hat{A}$ 的维数比 A 低吗,怎么可以做差值运算呢,公式的定义是不是有问题?这一点我们可以通过图 15.7 来解释,拟合直线本身是一维的,但它存在于二维空间。矩阵的原理相同,所以虽然 $\hat{A}$ 的阶数比矩阵 A 小,但它是通过轴变换表示在较低维空间上的,对于选择的特定轴,它应该和原矩阵具有相同的行列数。

针对这个问题,奇异值分解的任务是把词条-文档关系矩阵分解为三个矩阵的乘积形式^①,即 $T_{t \times n}, S_{n \times n}, D_{d \times n}$:

$$A_{t \times d} = T_{t \times n} S_{n \times n} (D_{d \times n})^T \quad (15.12)$$

其中, $n = \min(t, d)$ 。

例如,对图 15.5 中的矩阵 A 进行分解,矩阵 A 中的每一列实质上都表示一个文档向量,其中元素 a_{ij} 表示词条 i 在文档 j 中出现的次数,当然也可以用 1.2 节讨论的内容来计算这个值,但为简单起见,假设每个词条 i 出现的频度都为 1,暂不计算词条权重。

图 15.8 和图 15.10 分别给出了矩阵 T 和 D 的值,矩阵 T 和 D 的列都是正交关系。矩阵 S 是对角矩阵。如前所述,奇异值分解其实是通过旋转空间的轴得到的,它优先选取那些投影方差大的轴方向。 T 和 D 分别代表了新空间中的词条和文档矩阵,道理很简单,比如矩阵 T 中的列对应 A 中的行,因为 T 可以表示成 $A * X$ 的形式, A 中的行表示词条。

对角化矩阵 S 的对角线元素实际上是 $A^T A$ 的特征值(见图 15.9),并且按照降序顺序排列,第 i 个特征值表示在第 i 个投影轴上的偏差,我们只取矩阵 T, S, D 的前 k 列,得到矩阵 $T_{t \times k}, S_{k \times k}, (D_{d \times k})^T$ 。从数学上可以严格证明,一个矩阵的 SVD 分解结果是惟一确定的^②(Golub and van Loan, 1989)。

找到 SVD 降维分解的最优映射是分析词条同现的关键,在这个过程中具有相同同现模式的词条被映射到同一维上,例如我们取图 15.8 中矩阵 T 的前两维,从图中可以看出词条在第二维上有明显的区分:单词 *cosmonaut*, *astronaut* 和 *moon* 都是航天领域的词汇,它们的值都是负值,而单词 *car* 和 *truck* 是交通工具领域的单词,它们都取正值,所以可以看出在第二个维度上组词的同现模式被区分得很清楚。这也带来了一个缺点,即有时候这类信息是个误导,但大多数情况下它还是基本反映了词条之间的内在相关性。

$$T = \begin{pmatrix} & \text{Dim. 1} & \text{Dim. 2} & \text{Dim. 3} & \text{Dim. 4} & \text{Dim. 5} \\ \text{cosmonaut} & -0.44 & -0.30 & 0.57 & 0.58 & 0.25 \\ \text{astronaut} & -0.13 & -0.33 & -0.59 & 0.00 & 0.73 \\ \text{moon} & -0.48 & -0.51 & -0.37 & 0.00 & -0.61 \\ \text{car} & -0.70 & 0.35 & 0.15 & -0.58 & 0.16 \\ \text{truck} & -0.26 & 0.65 & -0.41 & 0.58 & -0.09 \end{pmatrix}$$

图 15.8 图 15.5 的矩阵奇异值分解中的 T 矩阵(其中的值已进行了四舍五入)

- ① 从技术上讲,这应该是简化 SVD 的定义。完全 SVD 的形式是 $A_{t \times d} = T_{t \times t} S_{t \times d} (D_{d \times d})^T$ 。 S 中多余的行和列是零向量, T 和 D 是正交方阵(Trefethen and Ball 1997:27)。
- ② 对于给定的奇异值分解结果,可以通过改变相应的左右奇异向量 T 和 D 的符号来得到不同的解。而且,如果有两个或者两个以上的相同的奇异值,那么由对应奇异向量决定的子空间就是惟一的,但是可以通过不同的正交基来描述。除了这些例子以外,SVD 一般都是惟一的。

$$S = \begin{pmatrix} 2.16 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.59 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.28 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.39 \end{pmatrix}$$

图 15.9 图 15.5 的矩阵奇异值分解中的奇异值矩阵(其中的值已进行了四舍五入)

$$D^T = \begin{pmatrix} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \text{Dimension 1} & -0.75 & -0.28 & -0.44 & -0.45 & -0.33 & -0.12 \\ \text{Dimension 2} & -0.29 & -0.53 & -0.19 & 0.63 & 0.22 & 0.41 \\ \text{Dimension 3} & 0.28 & -0.75 & 0.45 & -0.20 & 0.12 & -0.33 \\ \text{Dimension 4} & 0.00 & 0.00 & 0.58 & 0.00 & -0.58 & 0.58 \\ \text{Dimension 5} & -0.53 & 0.29 & 0.63 & 0.19 & 0.41 & -0.22 \end{pmatrix}$$

图 15.10 图 15.5 中矩阵分解后矩阵 D 的转置矩阵(数值进行了四舍五入)

这类词条之间的相似度对文本相似度的判断有很大的作用,设想降维为两维空间后的情形,经过奇异值调整,我们得到了矩阵 $B = S_{2 \times 2} D_{2 \times d}^T$,如图 15.11 所示,其中矩阵 $S_{2 \times 2}$ 已经被严格约束到了二维(对角元素为 2.16, 1.59)。矩阵 B 是原始矩阵 A 降维后的结果,如图 15.6 所示。

	d_1	d_2	d_3	d_4	d_5	d_6
Dimension 1	-1.62	-0.60	-0.04	-0.97	-0.71	-0.26
Dimension 2	-0.46	-0.84	-0.30	1.00	0.35	0.65

图 15.11 $B = S_{2 \times 2} D_{2 \times d}^T$ 分解后并降到二维的结果(数值进行了四舍五入)

表 15.9 中列出了文档在新空间中的相似度, d_1 和 d_2 之间的相似度为 0.78, d_4 , d_5 和 d_6 为 0.94, 0.93 和 0.74,而在原空间上两者的值是相等的。在原空间中, d_2 和 d_3 这两篇文档中没有共同的单词,相似度为 0,但是在新空间中的相似度为 0.88,之所以有这种结果,在于它们之间存在着同现模式。

注意,如果在没有进行降维处理的转换空间中计算相似度,就会得到和原空间中相同的相似度(即相似度为零)。使用图 15.10 中的向量,并用合适的奇异值改变维数,即可得到:

$$\begin{aligned} & -0.28 \times -0.20 \times 2.16^2 + -0.53 \times -0.19 \times 1.59^2 + \\ & -0.75 \times 0.45 \times 1.28^2 + 0.00 \times 0.58 \times 1.00^2 + 0.29 \times 0.63 \times 0.39^2 \approx 0.00 \end{aligned}$$

如果实实在在地计算一遍上述公式,会发现结果并不为 0,但这是由于四舍五入的误差造成的。实际上,矩阵运算对于四舍五入误差也是非常敏感的。

表 15.9 文档相关性矩阵 $E^T E$, 矩阵 E 由 B 经过列长度归一化得到。例如文档 d_3 和 d_2 (如图 15.11 所示) 的归一化相关系数为 0.88 (其中的值已进行了四舍五入)

	d_1	d_2	d_3	d_4	d_5	d_6
d_1	1.00					
d_2	0.78	1.00				
d_3	0.40	0.88	1.00			
d_4	0.47	-0.18	-0.62	1.00		
d_5	0.74	0.16	-0.32	0.94	1.00	
d_6	0.10	-0.54	-0.87	0.93	0.74	1.00

降维空间中文档相似度是通过矩阵 D 和 S 来计算的,之所以可以这样计算,是因为在原空间文档之间的相似度实际上可以由计算 $A^T A$ 得到。

$$A^T A = (TSD^T)^T TSD^T = DS^T SD^T = (SD)^T SD^T \quad (15.13)$$

矩阵 T 为正交矩阵,同理,下式可以用来计算词条之间的相似度:

$$AA^T = TSD^T(TSD^T)^T = TSD^T DS^T T^T = (TS)(TS)^T \quad (15.14)$$

但是,如何在降维空间中表示查询字段和新增文档,仍然是个在实际应用中有待解决的问题,因为 SVD 的方法仅仅给出了分解矩阵 A 的方法。如果每次新查询都重新计算 SVD,显然会带来计算量上的问题;另外,对于大的语料库,我们只对其中一些文档做了 SVD 分解,而希望后续的文档通过使用前面的计算结果进行转换。

$$\begin{aligned} A &= TSD^T \\ \Leftrightarrow T^T A &= T^T TSD^T \\ \Leftrightarrow T^T A &= SD^T \end{aligned} \quad (15.15)$$

上面的公式就是解决方法,我们只需进行查询字段、文档向量以及降维后的词条矩阵 T 之间的运算,例如有一个新查询向量 $\vec{q}$ 和维数 k (降维后),那么 $\vec{q}$ 在降维后的新空间中表示为 $T_{t \times k}^T \vec{q}$ 。

15.4.3 信息检索系统中的潜在语义索引

SVD 在信息检索领域的应用最初是 20 世纪 90 年代由一些研究者(Deerwester et al. 1990)提出来的,他们对潜在语义索引和传统的向量空间法做了一些比较,发现在很多情况下 SVD 的性能都比向量空间法更好,特别是在要求有高召回率的情况下(Deerwester et al. 1990; Dumais 1995)。道理其实不难理解,因为潜在语义索引利用了同现度;但是,在另一方面,引入的假相关也带来了准确度的下降。

潜在语义索引的应用依赖于具体的文档集合,比较适用于词汇异构度很高的文档集合。例如,在图 15.8 中,文档中采用不同的词汇来阐述同一个问题,但是如果文档中的词汇异构度较低,应用潜在语义索引的效果将不会太明显。

正如我们所见,这个方法之所以称为潜在语义索引,是因为把文档在一个新的降维空间中表示,新的降维空间实际上是原有空间的各维的线性组合,并且认定新空间才真正表示出了文档的内在联系,这也是所有的空间降维方法的假设前提。在实际情况下,文档生成过程扰乱了这个真正空间,使位于某个特定维度上的单词集合一部分分布在一些文档中,一部分分布在另外一些文档中。而潜在语义索引揭示了原有空间上的语义结构。我们可以把这种将不同单词映射到同一个潜在维度上的方法看成是一种软聚类的方法,因为它是按词条在降维空间的维来实现组合的。

在揭示内在语义联系的同时,LSI 还体现出了紧凑的优点,因为很多稀疏空间向量的维数远远大于 150,而潜在语义索引通常可以降到 150 维左右。但是,潜在语义索引的速度比传统方法慢,因为算法本身需要进行高阶矩阵的计算,计算查询字段和每一篇文档的相似度。

潜在语义索引的计算复杂度大概为词条-文档矩阵的阶数的平方,或者是特征值个数的立方这一量级(Deerwester et al. 1990:395)^①。对于非常大的文档集合,选取它们的子集和出现频

① 但是,其他一些研究者认为,当把 SVD 方法应用于信息检索问题时,由于矩阵的特殊属性,算法的复杂度和文档数目之间为线性关系,大概为奇异值数目的平方这一量级。详细的讨论可以参考(Oard and DeClaris 1996),(Berry et al. 1995)和(Berry and Young, 1995)。

率很高的词汇是降低计算量的一个有效方法。

但是奇异值分解也有局限性,和其他最小平方和方法相同,它假设数据的分布是正态分布的。如果读者阅读了本书前面的章节,就应该知道类似词频这样的统计量并不符合正态分布的条件。最小平方和方法 and 正态分布之间的联系可以通过下面的正态分布公式来了解(见 2.1.9 节):

$$n(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp \left[-\frac{1}{2} \left(\frac{x - \mu}{\sigma} \right)^2 \right]$$

其中 μ 是均值, σ 是方差。 $(x - \mu)^2$ 的值越小, $n(x; \mu, \sigma)$ 概率值就越大,所以我们说最小平方和法从本质上来说就是最大似然概率法,但它的前提是假设数据的分布是正态分布的。正态分布其实不适合信息检索的数据,其他分布,如负二项分布和泊松分布,对于计算词条出现的这类数据更有效。还有一个问题,既然我们假设数据是正态分布的,那么就允许出现负值,在前面的例子中我们已经看到了这种情况,但在基于泊松分布的维数消减中这种情况是不可能出现的。

从实际的应用角度来看,潜在语义索引方法的计算量太大,和其他利用词条同现关系方法相比,其性能并没有提高很多,而一些很简单的方法却能取得很好的性能,比如伪反馈(pseudo-feedback, pseudo relevance feedback, two-stage retrieval)方法(Buckley et al. 1996; Kwok and Chan 1998),其思路相当简单。该方法将 ad-hoc 检索得到的前 n 个文档(通常为 10 或者 20 篇)的关键字段信息加入到查询字段中,虽然不排除其中一些文档和查询是不相关的,但是取 10 或者 20 篇文档,就能确保我们能够获得足够的和查询字段同现的词汇。所以,伪反馈的方法在某种程度上可以看成是一种简单的基于同现分析的方法。

但是,总的来说,和许多加入启发信息的利用统计词条出现频度的方法相比,潜在语义索引的方法具备框架定义完整,优化准则清楚的特点。正是由于定义良好、思路清晰的特点,使潜在语义索引研究成为信息检索领域中一个备受关注的方法。

15.5 篇章分割

如今文档集合的异构程度越来越高。表现得最明显的就是文档的长度,尤其是互联网上的文档,它们之间的长度差别很大。在信息检索系统中有关如何处理文档长度的问题在 15.2.2 节中已经做了介绍,主要是采用余弦归一化的方法,但是余弦法归一化和其他归一化方法都是针对词条出现的频率和文档的长度做均值化处理,并没有考虑到词条的实际分布情况。假定去寻找一段关于“angioplasty”的简短描述,它最可能集中出现在一篇文档的几个段落中,这也是用户期望找到的,因为这样的段落中常常包含 angioplasty 是什么的定义。而如果这个单词在一篇文档中出现的位置很分散,通常这篇文档并不是用户所需要的。

篇章分割的主要任务在于通过对文档结构进行分析,寻找和查询有关的段落,并把找到的段落而不是整篇文档返回给查询用户。但是,很多文档没有明显的段落标记,所以必须找到一种方法将文章分段,每一段都涉及相同的主题内容。因为文档结构不同,有些文档本身有很清楚的结构,而另外一些则完全没有诸如段落、章节之类的信息。如何理清它们的结构,按照主题分割文档,定位用户感兴趣的内容,才是文档分割要致力解决的问题。这类方法通常称为 TextTiling 算法(Hearst and Plaunt 1993; Hearst 1994, 1997)。

15.5.1 TextTiling 算法

TextTiling 算法的基本思想是在一篇文档中寻找从一个主题转到另一个主题的“过渡”部分。因为文档中的各个句子长度不同,可能差别很大,所以首先把句子划分成固定长度的子句(token sequence)。一般子句的长度取 20 个词,子句之间的点称为间隔点(gap)。一般的分割算法主要由三个部分组成:紧凑度计算(cohesion scorer)、深度计算(depth scorer)和边界选择(boundary selector)。

紧凑度计算部分衡量主题内容在间隔点的连续性。不言而喻,我们希望找到紧凑度低的间隔点,紧凑度低意味着前后的连续性差,可以作为分割的候选点。

深度计算部分的作用是,对每一个间隔点,将它的紧凑度和周围的间隔点进行比较,相对值越低,那么它的深度值就越大。如果紧凑度和周围间隔点的值差不多相等,那么它的深度值就低。所谓的紧凑度概念是相对的,在一些文本中它的内容的主题可能变化很大,例如文档的引言(introduction)部分,其中覆盖了文档的全部内容。我们关心的是那些紧凑度最低的间隔点。而与此相对应的是,一篇文章可能一连几页的主题变化都很细微,这时的明智选择是那些虽然紧凑度值比较高,但是和周围的点相比值又偏低的点,即采用深度算法。

边界选择部分的目标是通过比较深度值确定切割的候选点。

下面是几种计算紧凑度的方法。

- **向量空间计分法(Vector Space Scoring)** 方法很简单,第一步确定间隔点,分别将间隔点左右相邻的子句表示成向量的方法,转化的方法在前面的章节中已经做了阐述。然后,计算向量之间的相关系数。相关系数与切割点的概率成反比,相关系数越大,说明前后的紧密程度越高。这个方法最早是由 Hearst and Plaunt(1993)提出,由 Salton and Allen(1993)首先使用的。
- **文本块比较法(Block Comparison)** 文本块比较方法的思路和向量空间法相同,惟一的不同点是计算它的词条的权重时只利用文本块内词条的出现频度,而不利用文档频度。相关试验表明文本块比较法的效果最好。
- **新词引入法(Vocabulary Introduction)** 这种算法中紧凑度的值和文本块中的新词的个数成反比,即认为文本子主题的变化是通过新词的引入进行的。在实际的计算中我们只需要将新词的个数乘以 -1 作为紧凑度的值。

Hearst(1997)所做的实验表明,文本块比较法在上述三种方法中效果最好。

TextTiling 算法的第二步是把紧凑度转化成深度值,具体计算方法是将当前间隔点和左右相邻的点高度差相加,作为深度值。例如,在图 15.12 的 text 1 中, g_2 点取 $(s_1 - s_2) + (s_3 - s_2)$ 。注意,紧凑度的高绝对值本身并不能导致分割边界的产生。TextTiling 算法认为,子主题的改变和分割是有关联的。在一个多段落文本中,不同段落的主题或词汇波动频繁,但是最基本的变化是和分割边界一致的。在处理子主题细微变化的文本中,这个算法将取得更好的识别效果。

对于实际上的算法实现,需要对基本算法进行一些增强。首先,我们要对紧凑度进行平滑,以应用到如图 15.12 的 text 2 所示的情形中。从直觉上来看, $s_1 - s_2$ 应该对 g_4 点深度值的计算有所贡献,这是通过使用一个低通滤波器对深度值进行平滑得到的。例如, g_i 的深度值 s_i 将被 $(s_{i-1} + s_i + s_{i+1})/3$ 替代。这个过程有效地考虑了某些间隔点的紧凑度,这些间隔点离中

心间隔周围的两个间隔点有一段距离。如果它们和两个直接包围间隔的值同样高甚至更高,就会增大中心间隔的值。

我们还需要加入启发式算法,以避免产生许多小片段(当把文本分成连贯的单元时,很少选择这种分割方法)。最终,计算紧凑度和深度值的方法的参数(词次序列的大小、块的大小和平滑方法)可能需要依赖于我们正在处理的文本类型进行调整。例如,包含长句的语料库需要长的词次序列。

TextTiling 算法的第三个部分是边界选择器。它估计深度值的均值 μ 和方差 σ ,选择所有深度值高于 $\mu - c\sigma$ (c 为常数,例如 c 为 0.5 或 1.0)的间隔作为边界。我们再次避免了使用绝对值。这个方法选择那些有“重要意义”的低深度值的间隔,这里的“重要意义”的定义是和值的均值和方差相关的。

通过评测,Hearst(1997)发现这个算法所确定的分割和人工判断的分割吻合得相当好。用基于准确率和召回率来评测,与文本检索相比,在多大程度上分割检索会导致更好的信息检索性能,仍然是一个开放的问题。然而,许多用户更愿意看到一个文本自然分割的上下文的成功描述,这样的上下文更容易被很快地理解(Egan et al. 1989)。

文本分割在自然语言处理的其他领域也有重要应用。例如,在词义消歧中,文本分割可以用来找到对于确定一个用法的正确语义具有最大信息量的自然语言单位。在差异不断增加的文本集合中,文本分割依旧是统计语言处理和信息检索中的一个重要研究课题。

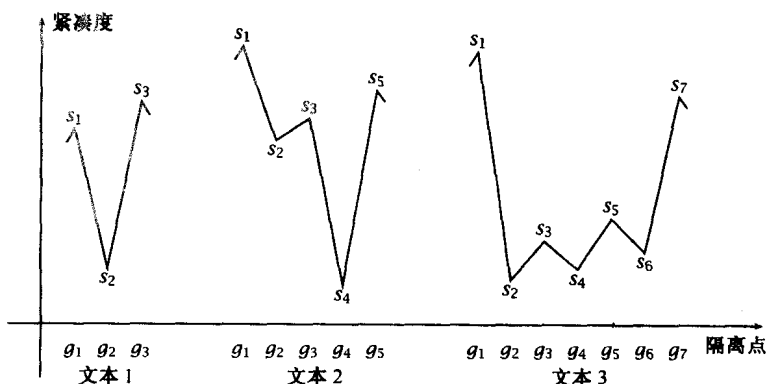


图 15.12 三个主题边界界定中的紧凑度值分布图

15.6 深入阅读

在信息检索领域中,TREC 会议的出版物以及 ACM SIGIR 都是相当好的参考资料,其他一些有关信息检索的比较有名的期刊包括: Information Processing & Management, The Journal of the American Society for Information Science, Information Retrieval。

信息检索的书籍有很多,下面我们推荐几本比较好的书籍。van Rijsbergen(1979), Salton and McGill(1983)以及 Frakes and Baeza-Yates(1992)对信息检索的阐述都比较清楚,也可参见(Losee 1998)和(Korfhage 1997)。Sparck Jones and Willett(1998)收集了一些信息检索研讨会的论文。Smeaton(1992)以及 Lewis and Sparck Jones(1996)讨论了信息检索在自然语言处理领域中所扮演的角色。(Cleverdon and Mills, 1963), (Tague-Sutcliffe 1992)和(Hull 1996)对信息检索系统的评估问题进行了阐述。Sparck Jones(1972)最早提出将倒排文档频度方法用于计算词条权重。

在康奈尔大学, Gerard Salton (Salton 1971b; Salton and McGill 1983) 在 SMART 项目中系统研究了 tf.idf 权重的不同计算方法。关于这方面的最近的文献, 读者可以参阅 (Singhal et al. 1996) 和 (Moffat and Zobel 1998)。

关于泊松分布的资料, 读者可参看具体的概率论书籍, 比如 (Mood et al. 1974:95) 和 (Harter 1975) 都详细讨论了二重泊松分布的参数 π , λ_1 和 λ_2 的估计问题。本章对权重计算部分的介绍是依照 (Croft and Harper 1979) 和 (Church 1995) 为基础展开的。

近年来, 实际上除了短语抽取算法, 自然语言处理对信息抽取研究基本上没有大的贡献, 因为大多数信息检索的研究都集中于浅层分析技术。当然, 有一些研究者基于自然语言处理对信息检索做了比较深入的探讨, 例如 (Fagan 1987; Bonzi and Liddy 1988; Sheridan and Smeaton 1992; Strzalkowski 1995; Klavans and Kan 1998), 但是大部分内容都是集中于自动文本摘要方面 (Salton et al. 1994; Kupiec et al. 1995)。

自然语言处理对信息检索的一个有用之处是跨语信息检索系统 (cross-language information retrieval) 的建立, 即 CLIR 系统 (Hull and Grefenstette 1998; Grefenstette 1998)。跨语信息检索系统适合在一定程度上理解某种外语但又不很精通的用户查询外语资料, 最近这方面的研究有 (Sheridan et al. 1997; Nie et al. 1998), AAAI 研讨会关于跨语和语音检索的专题参见 (Hull and Oard 1997), CLIR 系统的潜在语义索引参见 (Littman et al. 1998a; Littman et al. 1998a)。

本章只对词条模型做了很简单的阐述, 读者如果需要阅读更深入的内容, 可以参阅 (van Rijsbergen 1979:ch.6), (Robertson and Sparck Jones 1976) 和 (Bookstein and Swanson 1975)。此外, 信息论的知识也可以用来提高信息检索的结果, 参见 (Wong and Yao 1992)。(Yamamoto and Church 1998) 给出了如何用 RIDF 建立索引的方法。

表 15.8 摘自 (Deerwester et al. 1990), 我们的词条-文档矩阵规模很小, 可以用相关的统计软件来分解。当然, 在实际应用中, 特别是在大语料库条件下, 必须采用特别的方法提高其计算效率 (Berry 1992)。

SVD 的方法除了应用于词条-文档矩阵的分解之外, 还用于词-词矩阵分解 (Schütze and Pedersen 1997), 也用于文本分割 (Kaufmann 1998)。Dolin (1998) 采用潜在语义索引的方法进行分布式搜索和查询分类的研究。

潜在语义索引本身可以作为一个人类记忆感知的描述模型, Landauer and Dumais (1997) 利用潜在语义索引说明学龄儿童的词汇量迅速增加的原因。

目前篇章分割也是一个研究热点, 这方面的研究包括 (Salton and Buckley 1991), (Beeferman et al. 1997) 和 (Berber Sardinha 1997)。Kan et al. (1998) 在网上公布了这些算法的实现程序, 有兴趣的读者可以查询。Text Tiling 算法利用了称为“词链” (lexical chains) 的信息, 它们是一系列在语义上相关的词汇。通过观察这些词汇可以大致推导出文本的子结构 (Morris and Hirst 1991)。

篇章分割算法只是对文档结构的一个粗划分, 因为文档本身的结构其实很复杂, 完全识别其结构是很艰巨的任务。如果要进行深入的分析, 读者可以参阅 (Walker and Moore 1997), 它做了很好的介绍。

当统计自然语言处理在 20 世纪 90 年代成为研究热点时, 篇章建模 (discourse modeling) 方面的研究相当少, 它成为热点只是近几年的事情。对话建模的文献可以参见 (Stolcke et al. 1998), (Walker et al. 1998) 和 (Samuel et al. 1998)。指代消解 (anaphora resolution) 可参见 (Kehler 1997) 和 (Ge et al. 1998)。

15.7 习题

- 15.1 [★]总结现有的各种搜索引擎的特点。例如,搜索引擎中是否利用了停用词表?试着在查询字段中输入一些停用词表中的单词,如输入短语 *the the* 能否查询?搜索引擎是否利用了词干化算法?是否区分大小写,比如,输入 *iNfOrMaTiOn*,观察返回值。
- 15.2 [★]在向量空间模型中,最简单的短语处理方法是将其划分为单个词条进行处理。例如,查询字段 *car insurance rates* 可以分解为单独的词条 *car*, *insurance*, *rates*, *car insurance* 和 *insurance rates*,说明其是否合理?
- 15.3 [★]证明 Katz 的 k 重混合模型满足:

$$P_i(0) = 1 - \frac{df_i}{N}$$

即词条在一篇文档中没有出现的概率估计和出现该词条的文档数相对应。

- 15.4 [★]计算表 15.7 中的剩余倒排文档频度,说明其是否能正确区分出实词和虚词。
- 15.5 [★]任意选择一个虚词、一个实词和一个未知类别信息的单词,计算下面各个数量值:
(a) 文档频度和收集频度, (b) IDF, (c) RIDF, (d) k 重混合模型中的参数 α 和 β 值(你可以选择一个适当规模的语料库)。
- 15.6 [★]我们知道泊松分布的参数 λ 不同,可能使泊松分布单调下降,或者首先上升然后下降,举例说明参数 λ 对泊松分布的影响。
- 15.7 [★]采用 SVD 的方法分解图 15.5 中的词条-文档矩阵,读者可以利用 S-Plus 或其他统计软件工具。
- 15.8 [★★]TextTiling 算法中关于子主题结构有两个基本假设,TextTiling 算法本身是一个线性分割算法,这是第一个假设前提,也就是说,算法把文本划分为连续文本段。Hearst 说明了一个反例,假设当前有四个连续文本段,第四段是前三个段落的总结。那么这四段从理论上讲应该作为一个整体,但是如果前两段中的单词没有一个出现在第三个文本段中,有一些出现在第四段上,TextTiling 算法会将第三个文本段和第四个文本段分割开来。提出一种对 TextTiling 算法的修改方法,使之能把一至四段文本识别为一个整体。第二个假设是 TextTiling 分割算法依赖于类似图 15.12 中的曲线“谷底”的存在,但是有时候两个段落之间可能是平滑过渡,在曲线中表现为一个平坦的长区域,这也是算法本身的一个缺陷,试给出解决这个问题的办法。

第16章 文本分类

本章阐述自然语言处理中的一个重要应用领域,即文本分类(text categorization)问题,并对分类问题进行概要性展望分析,包括一些在本书其他章节中没有涉及但相当重要的分类判定算法。顾名思义,所谓分类就是把来自于同一总体的对象划分成两个或更多的类别概念。表 16.1 给出了一些例子。本章之前我们已经详细讨论过的词性标记、语义消歧、介词短语附着都属于分类问题。根据词的上下文信息,我们可以判定它所属的词性或语义类别;在介词短语附着上,所谓两类问题是指两种不同的介词附着形式;文章作者和所用语言的识别也是自然语言处理中经常需要处理的分类问题,例如,有时需要判定一首诗的作者是莎士比亚,还是出自其他人的手笔,我们称这类问题为作者识别问题(author identification),语言识别则是解决一个文档的原始出处的问题(见习题 16.6)。

表 16.1 自然语言处理中的一些分类问题,针对每个问题
表中都给出了其适用的对象和可能的类别信息

问题	处理对象	类别
标注	词的上下文	词的标记
消歧	词的上下文	词的语义
介词短语附着	句子	句法分析树
作者识别	文档	作者
语言识别	文档	语言
文本分类	文档	文档主题

本章重点讨论文本分类问题。文本分类的目的是把文档按照它们的主题或者主旨来划分类别。其中,我们曾经用过的路透社文档集合是一个很有代表性的按照文章主题进行分类的语料库,其中它的主题包括“兼并和收购”、“小麦”、“原油”、“收入报告”等,可以按照内容将文章划分到各自所属的类别下。文本分类应用很广,其中一个重要应用是新闻过滤,即筛选出读者感兴趣的内容。比如,在上面的几类文档中金融记者可能只需要“兼并和收购”类别的新闻。

文本分类系统的任务可简单定义为:给定分类体系后,根据文本内容自动确定文本关联的类别。从数学角度来看,文本分类是一个映射过程,它将未标明类别的文本映射到现有类别中,该映射可以是一一映射,也可以是一对多映射,因为通常一篇文本可以与多个类别相关联。

文本分类的映射规则是,系统根据已知类别中若干样本的数据信息总结出分类的规律性,建立类别判别公式和判别规则。当遇到新文本时,根据总结出的类别判别规则确定文本所属的类别。

一般来说,统计分类问题的形式化表述如下。在特定训练集合中,每一个实例都被赋予一个或多个模式类别,可以用数据表述模型(data representation model)表示。训练数据集合中的每个数据以 $(\vec{x}, c)$ 来表示, $\vec{x} \in \mathbb{R}^n$,其中 $\mathbb{R}^n$ 是数据的向量形式, c 是类别标记。对于文本分类问题,向量空间模型是算法中经常采用的数据表达方式。也就是说,首先每一篇文档都表述成词频的向量形式(参见 15.2 节),然后定义算法的模型类别(model class)和具体训练过程(training procedure),模型类别实际上是参数化的分类器族(family of classifiers),而训练过程就是从分

类器族^①中选择具体的一个分类器算法的过程。例如,二值分类问题的决策面实际上是如下形式的线性分类器:

$$g(\vec{x}) = \vec{w} \cdot \vec{x} + w_0$$

当 $g(\vec{x}) > 0$ 时样本属于类别 c_1 , 当 $g(\vec{x}) \leq 0$ 时样本属于类别 c_2 , 其中参数 $\vec{w}$ 和阈值 w_0 决定了分类器的具体形式。

实际上训练算法和函数拟合过程相当类似,都是按照某种最优化原则寻求合适的参数集合,比如这个优化原则是基于分类错误率或者是基于熵的。当然,其中一些训练方法能确保全局最优解,但是很多迭代训练过程仅仅能够保证局部最优解。特别值得指出的是,如果初始的参数空间假设是错误的,那么找到全局的最优解是不可能的,典型的例子就是应用梯度下降法 (gradient descent) 或者爬山法 (hill climbing) 的线性分类器 (在本章的稍后章节将详细讨论)。

确定了分类器的训练参数后,评估分类器的性能是很有必要的。测试数据集合不应该包括训练数据,因为基于训练数据的测试结果并不具备说服力,真正的测试应该选取具有代表性的未知样本集合,否则算法的真实性能就无从谈起。

对于二值分类问题,分类器一般采用表 16.2 中的统计数据来评估其分类性能。一个很重要的度量是正确率 (accuracy), 我们表示成 $\frac{a+d}{a+b+c+d}$, 即分类正确的实例所占的比例。另外,度量还有准确率 (precision) $\frac{a}{a+b}$; 召回率 (recall) $\frac{a}{a+c}$; 错误误识率 (fallout) $\frac{b}{b+d}$ 。参见 8.1 节。

表 16.2 两类问题的分类评估关联表。例如, a 表示在感兴趣的类别中的数据对象被正确分类的数目

	YES is correct	NO is correct
YES was assigned	a	b
NO was assigned	c	d

实际算法评估过程中常用的两个指标是准确率和召回率。准确率和召回率反映了分类质量的两个不同方面,两者必须综合考虑,不可偏废。因此,存在着一种新的评估指标, F_1 测试值, F_1 测试值 = (准确率 * 召回率 * 2) / (准确率 + 召回率)。

对于类别判定问题,特别是在多类的情况下, 2×2 的关联表 (contingency table) 是经常采用的衡量算法性能的一种手段,类别和表一一对应。表的计算有两种方法,第一种方法是针对每个表计算诸如识别率这样的评估指标,然后得出其平均值,这种方法称为总体均值法 (macro-averaging); 与此相反,另一种方法称为个体均值法 (micro-averaging), 即制定一个单一的大表,在每个表单元中统计所有类别的数据信息的和值,然后计算这个单一的大表的度量值。两种方法的不同点在于,总体平均法中赋予每个类别相同的权重值,个体平均法中则赋予每个文档对象相同的权重值。当计算准确率的加权均值时,两种方法计算出来的结果是不同的。个体平均法倾向于含有元素数目比较多的“大类别”,而总体平均法则反映了分类器在所有类别上的分类效果。

本章我们主要介绍四类分类算法,分别是决策树、最大熵建模、感知器方法和 k 最近邻法。它们都是很重要并且应用很广泛的分类算法。在感知器方法中,我们只介绍其最简单的神经网络形式,如果读者对更深入的内容感兴趣,可以参阅本章最后的深入阅读小节。

^① 需要读者注意的是,一些分类器 (例如最近邻法等一些非参数的分类器) 很难被划归为某一个模型类型。

16.1 决策树

我们首先介绍决策树分类模型。图 16.1 是决策树的一个例子,它判定一篇文档是否属于路透社文档集中定义的 *earnings* 类别。分类过程从树的根节点开始,自顶向下,根据当前文档属性在每个树节点对当前问题的回答,选择不同的树分支,重复迭代,一直到树的叶节点。总体来说,决策树学习着眼于从一组无次序、无规则的事例中推理出决策树形式的分类规则。它采用自顶向下的递归方式,在决策树的内部节点进行属性值的比较,并根据不同的属性值判断从该节点向下的分支,在决策树的叶节点得到结论。所以,从根到叶节点的一条路径就对应一条合取规则,整棵决策树就对应着一组析取表达式规则。基于决策树的学习算法的一个最大优点是,它在学习过程中不需要使用者了解很多背景知识(这同时也是它的最大缺点),只要训练例子能够用属性-结论式的方式表达出来,就能使用该算法来学习。例如,在图 16.1 的决策树中,当一个文档的 *cts* 属性权重等于 1 并且 *net* 属性权重等于 3 时,决策树首先根据 *cts* 属性值选取根节点的左子树,随后再根据 *net* 属性值选取右子树。概率值 $P(c|n_4)$ 给出了在树的节点 4 中属于 *earnings* 类别的概率为 0.649,也就是说,当文档的 *cts* 属性权重小于 2, *net* 属性权重大于等于 1 时,判断 $P(C|n_4)$ 属于 *earnings* 类别的概率为 0.649。图中决策树的每个节点表示的内容包括:训练数据集中属于该类的文档数目,属于 *earnings* 类别的概率,所用判别属性以及相应属性的权重阈值。

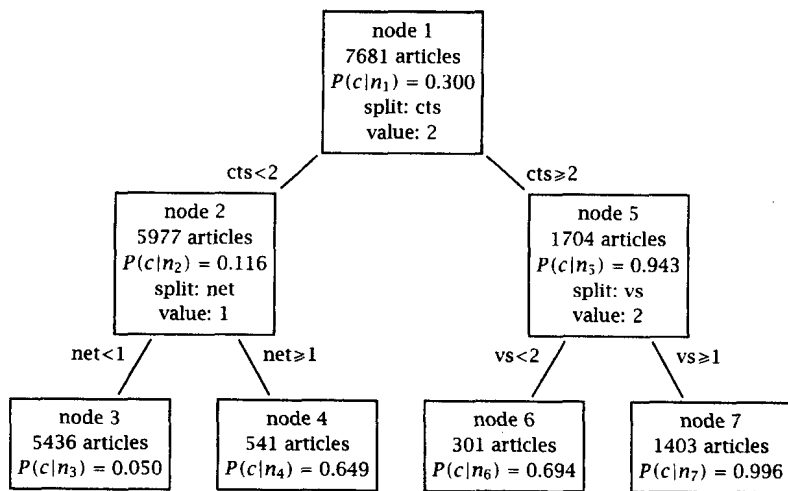


图 16.1 决策树。图中的决策树用于判定一篇文档是否属于 *earnings* 类别。 $P(c|n_i)$ 表示树节点 n_i 中的文档属于类别 *earnings* 的概率

图 16.2 是决策树的另外一种可视化的表示。图中水平轴对应于 *cts* 属性权重,垂直轴对应于 *net* 属性权重。决策树的问题是询问某个特征值是大于还是小于预定义阈值,如图 16.1 所示,决策树根节点所选取的判别属性对应于图 16.2 中垂直轴 $cts = 2$,树根的左节点将左面的区域依照 *net* 是否大于 1 划分成两个子区域,分别称为“上区域”和“下区域”(上下位置关系)。我们可以清楚地看出,在图 16.2 中,上区域(以 $P(c|n) = 0.649$ 标识)相当于树中的叶节点 4,下区域相当于树中的叶节点 3。这里需要注意的是,判定轴 $cts = 2$ 的进一步的划分并

不是按照 *cts* 属性,而是按照 *vs* 属性的阈值进行的。所以,如果想在图 16.2 中描述决策树中的节点 5,则必须引入三维空间表示的概念。

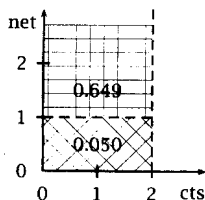


图 16.2 图 16.1 中的决策树片断的几何解释

本章的数据都来自路透社文档集合,我们的文本分类任务是将文档集合中属于 *earnings* 类别的文档挑选出来。实际上现在大多数文档分类研究都是以路透社语料库作为训练数据集的。我们所用的路透社语料库版本(Modified Apte Split, Apte'et al. 1994)中包括 9603 个训练文档,3299 篇测试文档,主要是 1987 年路透社的新闻资料。文档被分为 100 个主题,例如“兼并和收购”和“利率”等。图 16.3 给出了一些例子(可以在相关的网站查询到资料)。

```
<REUTERS NEWID="11">
<DATE>26-FEB-1987 15:18:59.34</DATE>
<TOPICS><D>earn</D></TOPICS>
<TEXT>
<TITLE>COBANCO INC &lt;CBCO> YEAR NET</TITLE>
<DATELINE> SANTA CRUZ, Calif., Feb 26 - </DATELINE>
<BODY>Shr 34 cts vs 1.19 dlrs
Net 807,000 vs 2,858,000
Assets 510.2 mln vs 479.7 mln
Deposits 472.3 mln vs 440.3 mln
Loans 299.2 mln vs 327.2 mln
Note: 4th qtr not available. Year includes 1985
extraordinary gain from tax carry forward of 132,000 dlrs,
or five cts per shr.
Reuter
</BODY></TEXT>
</REUTERS>
```

图 16.3 路透社新闻中的一个关于 *earnings* 主题的例子,为简明起见,此处省略了部分内容

如前所述,文本分类的首要任务是建立数据表示模型。建立数据表示模型实际上是一项技巧性很强的工作,依赖于使用的分类方法。本章中为了简化问题,统一采用一种数据表示方法:向量空间模型(Vector Space Model)。向量空间模型的基本思想是以向量来表示文本: $(W_1, W_2, W_3, \dots, W_n)$, 其中 W_i 为第 i 个特征项的权重,那么选取什么作为特征项呢?一般可以选择字、词或词组,根据实验结果,普遍认为选取词作为特征项更优于选取字和词组。最初的向量表示完全是 0 和 1 形式,即,如果文本中出现了该词,那么文本向量的该维为 1,否则为 0。这种方法无法体现这个词在文本中的作用程度,所以 0 和 1 逐渐被更精确的词频代替。词频分为绝对词频和相对词频。绝对词频,即用词在文本中出现的频率表示文本;相对词频为归一化的词频,其计算方法主要运用 *tf. idf* 公式。针对我们的分类问题,首先基于 χ^2 统计计算 *earnings* 类别的单词的得分,然后在训练集中取分值最高的前 20 个词作为文档的代表词。如表 16.3 所示,在 *earnings* 类中这些词包括 *loss*, *profit*, *cts* (for cents), 说明这三个词对文档内容是不是 *earnings* 类的指示作用很强。每一篇文档都表示成 20 维的向量, $\hat{x}_j = (s_{1j}, \dots, s_{Kj})$, 其中

S_{ij} 的计算公式如下^①:

$$s_{ij} = \text{round} \left(10 \times \frac{1 + \log(tf_{ij})}{1 + \log(l_j)} \right) \quad (16.1)$$

公式中 tf_{ij} 为词条 i 在文档 j 中的出现次数, l_j 为文档 j 的长度。 s_{ij} 为零的情况表示词条的出现次数为零。例如, 单词 *profit* 在一篇长度为 89 的文档中出现 6 次, 其 s_{ij} 值为 $s_{ij} = 10 \times \frac{1 + \log(6)}{1 + \log(89)} \approx 5.09$, 近似等于 5。与第 15 章中的方法相同, 计算采用了取对数的方法, 并考虑了权重因子的标准化。

表 16.3 图 16.3 所示文档 11 的详细表示, 阐明了本章中我们所用的数据表示模型的表述形式

单词 w^j	词条权重 s_{ij}	分类
vs	$\vec{x} = \begin{pmatrix} 5 \\ 5 \\ 3 \\ 3 \\ 3 \\ 4 \\ 0 \\ 0 \\ 0 \\ 4 \\ 0 \\ 3 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 3 \\ 2 \\ 0 \end{pmatrix}$	$c = 1$
mln		
cts		
;		
&		
000		
loss		
,		
"		
3		
profit		
dtrs		
l		
pct		
is		
s		
that		
net		
lt		
at		

图 16.3 中的文档的向量表示形式如表 16.3 所示。注意, 其中的一些单词, 例如单词 *that* 和 *s*, 其实对于文档分类的贡献率很小, 这是因为特征抽取过程采取了一些自动化方法, 而不是人工筛选方法, 这些很特殊的单词不可避免地被选入了。而其他三个标记 &, lt 和 ; 的选入是因为它们都是路透社文档特定的格式化符号。在 *earning* 类别中, 很大一部分文档的标题行都有 <CBCO> 形式的公司标志, 这个标志的左括号将被转化成 SGML 字符, 所以当遇到这种标记时, 很自然就会想到“这篇文档可能和某一个特定的公司有关”。很明显, 此种“元标记”(meta-tag)^② 对于判断类别大有益处, 从图 16.3 的例子中可以看出这一点, 在它的标题行中就有类似的标志。

现在我们有了一个决策树的类别判定模型和向量形式的数据表示方法, 很显然下面需要定义训练过程。一般决策树的训练过程是, 首先建立一个大规模的树结构, 然后对这个树剪枝到合理的尺寸大小。剪枝步骤是构建决策树必不可少的一个步骤, 因为大决策树通常都会出

① 本节中的对数都使用的是自然对数, 即以 e 为底。本节中的熵的计算也采用的是自然对数, 但是在其他章节中的熵的计算是以 2 为底的, 因此表 16.4 中的计算如果以 2 为底则数值分别为: 节点 1 的熵为 0.881, 节点 2 的熵为 0.518, 节点 3 的熵为 0.315。

② 字符串 < 本来应该作为一个词条来处理, 但是在文本分类中, 它可以作为一个经常出现的所谓的底层数据的例子。

现过适应(overfitting)问题,特别是当分类器基于训练集数据的一些相对比较次要的属性做出决策时,经常出现对训练数据的过适应,从而导致在测试集上的错误。例如,在比较极端的情况下,训练数据中只有一个文档,这个文档中包含单词 *dollars* 和 *percent*,但是这篇文档正好属于 *earnings* 文档类。显然,训练出的决策树当遇到具有这种属性的文档时,会将相应文档判定为 *earnings* 文档类。但是,正如前面所言,也许训练数据集中只存在一篇这样的文档,但是它却引起了决策树训练过度的问题。所以剪枝步骤必不可少,通过剪枝可以将决策树中类似的分支都删除,使决策树在训练数据上的性能更好。

针对决策树训练过程,我们需要制定相应的分支准则(splitting criterion)和停止准则(stopping criterion),分支准则决定树节点中选取何种属性作为当前训练数据集的分支属性,停止准则决定何时停止决策树的迭代过程。例如,一般的停止准则判断条件为:决策树的当前节点的所有数据都具备相同的数据表示形式或具有相同的类别,当前树节点的类别已经不能再做进一步的划分,这时就停止决策树的迭代过程。

本章中采用的停止准则是信息熵最大增益原则(maximum information gain)(Breiman et al. 1984:25; Quinlan 1986: Section 4; Quinlan 1993),即当按照某个属性分支信息增益最大时,将该属性作为决策树当前节点的决策属性。信息增益具体定义属于信息论的范畴,即父母节点熵值与子节点熵值加权之和之间的差值。

$$G(a, y) = H(t) - H(t|a) = H(t) - (p_L H(t_L) + p_R H(t_R)) \quad (16.2)$$

其中参数 a 为分支属性, y 为属性 a 的取值, t 表示当前处理节点的分布, P_L 和 P_R 表示左右子节点的数据元素在父母节点中所占的比例, t_L 和 t_R 表示左右子节点中数据元素的分布。例如,表 16.4 中表示了图 16.1 中顶层根节点的这些变量以及由这些变量形成的信息增益。

表 16.4 一个关于信息增益作为分支准则的例子。表中列出了图 16.1 中决策树的节点 1, 2 和 5 的熵值, 子树节点的权值和以及节点 1 分支为节点 2 和节点 5 的信息增益

节点 1 的熵, $P(C/N) = 0.300$	0.611
节点 2 的熵, $P(C/N) = 0.116$	0.359
节点 5 的熵, $P(C/N) = 0.943$	0.219
节点 2 和节点 5 的权值和	$\frac{5977}{7681} \times 0.359 + \frac{1704}{7681} \times 0.219 = 0.328$
信息增益	$0.611 - 0.328 = 0.283$

信息增益可以被解释成减少不确定性的一个度量,从直观上我们也感觉这种方法应该是有效的。原因在于信息增益最大导致了分支操作,也就是说,尽可能地减少了分类结果的不确定性。事实上,不存在通用的寻找最优划分值的有效算法,因此,在实际应用上,我们采用了一些启发式算法得到近似最优解^①。

按照停止原则,当迭代过程停止时,决策树的树节点停止分裂,形成叶节点。我们对叶节点概率值的预测基本上以叶节点上包含的数据集合为依据,计算它的最大似然估计,并进行适当的平滑计算。例如,如果一个叶节点上包含 6 个 *earnings* 类的文档以及其他的两个文档,那么我们可以按照以上原则计算一篇新的未知文档属于 *earnings* 类的概率大小: $P(\text{earnings} | d) =$

$$\frac{6+1}{2+6+1+1} = 0.7 \text{ (应用了加 1 平滑法,参见 6.2.2 节)}.$$

① 如果我们希望找到全局最优解,则可以进行穷尽搜索,因为 s_{ij} 是一个位于小区间内的整数。

当树构建完毕之后,需要对决策树剪枝,以优化其性能。与树的构建过程相似,对每个叶节点按照特定的准则评估其对分类器的作用,每次剪枝都保留那些分类判断贡献较小的叶节点(Quinlan 1993),因为贡献大的节点比较容易引起过适应的问题。剪枝也是一个迭代过程,其终止条件是直到无叶子节点可剪。其实剪枝的每一步都相当于得到了一个分类器,如何在这 n 棵(n 为初始决策树的内节点的数目)树中选取最好的一个是必须解决的问题,一种解决方法是利用留存数据验证。

验证(validation)过程是在留存数据(held-out data)上评测一个分类器的性能,即由验证数据集评估其分类正确率。正如要求测试集必须独立于训练数据,基于同样的原因,为了评估剪枝的程度,新的数据集作为剪枝操作的评估数据是必不可少的。具体方法参见 6.2.3 节的平滑方法。

另一个剪枝方法是首先保留整个决策树,但让决策树在计算叶节点分类概率的同时也给出决策树内部节点的概率值,这样无需剪枝就得出了决策树内部非叶节点的可靠数据分布。对于每一个叶子节点,我们只需读出从该叶节点到决策树根节点的所有树节点和相关的概率分布函数。与利用留存数据进行剪枝的方法不同,留存数据在此处用于训练线性插值的参数估计(参见 6.3.1 节),最后插值分布的结果作为最终的分类函数。Magerman(1994)认为这种方法的剪枝效果好得多,这一点至少从所研究的统计句法分析上得到了体现,参见 12.2.2 节。

图 16.4 表明决策树分类效果和剪枝操作的关系。X 轴代表剪掉的节点数, Y 轴代表分类正确率。我们将训练数据的 80%(7681 篇文档)作为训练集,剩余的 20%(1922 篇文档)作为验证数据集,图中没有显示根节点的剪枝过程^①。

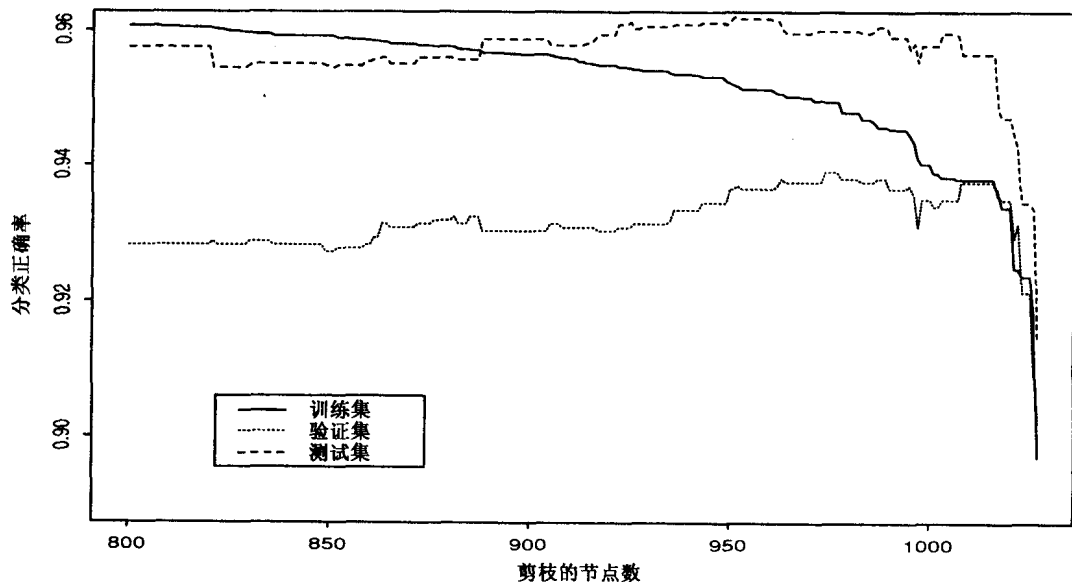


图 16.4 决策树剪枝。该图显示了决策树的分类正确率和剪枝之间的关系,在测试数据集上,当剪枝决策树的剪枝节点达到951个时(96.21%的正确率),决策树取得最优性能。而在验证数据上,当剪枝决策树达到最优性能时,剪枝数目大约在974至977左右(93.91%的正确率)。这4个剪过枝的树在测试数据上的正确率大约为96.00%,接近最优性能。此外,从图中可以看出,在训练数据上性能是单调递减的

^① 剪枝的准则为:在验证样本集中,需要剪枝的树节点的信息增益最小。

图中揭示的规律很简单:没有剪过枝的决策树的分类效果最佳。道理很简单,整个决策树的初始结构在训练集上取得最佳的效果,整个树比剪枝树更能拟合训练数据。从图中我们可以看出,从左至右决策树的性能逐渐变差。测试和验证集上的准确度在图形的中部有一个峰点,性能最好的峰点恰好是拟合训练集中次要属性的部分树被剪掉的点。简而言之,若要更进一步剪枝,将把那些正确反映属性和分类关系的树节点剪除,导致决策树性能的下降。

如何在剪枝后的决策树中选择一棵作为最终的结果,可行的策略是选取在验证数据集上性能最好的决策树。从图中可知,验证数据和测试数据的曲线是不一致的。在验证数据集上分类效果很好的决策树通常都稍微有些过适应或者训练不足,但和真实最优解差别不大。

表 16.5 评估了具有 50 个内部节点的决策树在测试集上的性能,它是在验证集上正确度能达到 93.91% 左右的规模最小的决策树。

表 16.5 路透社中 earnings 类别的决策树联立表。
在测试数据集上的分类正确率为 96.0%

分为 earnings 类别?	earnings 类别判断是否正确?	
	YES	NO
YES	1024	69
NO	63	2143

使用留存数据的一个问题是,相当一部分训练数据没有被使用,更好的办法是利用 n 重交叉验证(n -fold cross-validation)来估计决策树的剪枝程度(参见 6.2.4 节)。例如,在 5 重交叉验证中我们将数据分成 5 个部分。首先保留其中一部分作为验证数据,剩余的 4 部分数据用来训练决策树,然后用验证数据控制决策树的剪枝。这个训练过程重复 4 次。利用其结果初步估计出最优决策树的大小,最终的决策树是在整个训练数据集上训练的结果。学习算法的复杂度和在训练数据上的分类正确率是分类算法的核心问题。如果采用的算法的参数过多或者方法过于复杂,可能出现训练过适应问题,在新数据集上的效果得不到保证。反之,当训练算法过于简单,则可能会出现训练不足的情况,同样会导致正确率的下降。因此,找到它们之间的一个平衡点很重要,交叉验证不失为一个可取的方法。

分类算法的另一个常见问题是分类正确率和训练数据的相互依赖关系。毋庸置疑,训练数据越多,训练效果会越好,直到达到分类正确率和训练数据大小之间的一个均衡点。所以,也许有些偶然的因素导致我们在某个小训练数据下训练出的分类器效果相当好,但是这不等于说小训练数据就可以保证决策树的性能。

画出如图 16.5 所示的学习曲线对于确定合适的训练数据大小很重要。很多训练过程的计算量都很大,从这个角度讲,避免过大的训练数据集是合理的要求;但另一方面,训练数据不足同时又使决策树未达到最佳性能。

通过制定一个类似图 16.5 形式的曲线,可以大致估计出训练数据和分类算法之间的关系。当然,大多数情况下控制训练数据的规模大小是不可能的,虽然大的训练数据可以提高分类效果,但是获取它们具有相当的难度。

那么决策树算法适用于自然语言处理中什么样的分类问题呢? 决策树算法本质上比单纯贝叶斯(参见 7.2.1 节)、线性回归(参见 15.4.1 节)、逻辑回归(logistic regression)等分类算法复杂。如果分类问题本身很简单,例如线性可分问题,简单方法可能更适合。决策树训练过程是将训练数据一步一步划分成更小的子集,这个过程可能导致了算法的泛化性能下降,当然主要原因在于

训练数据不足。图 16.6 举出了一个简单的实例,这个实例是用来学习音韵规则的。

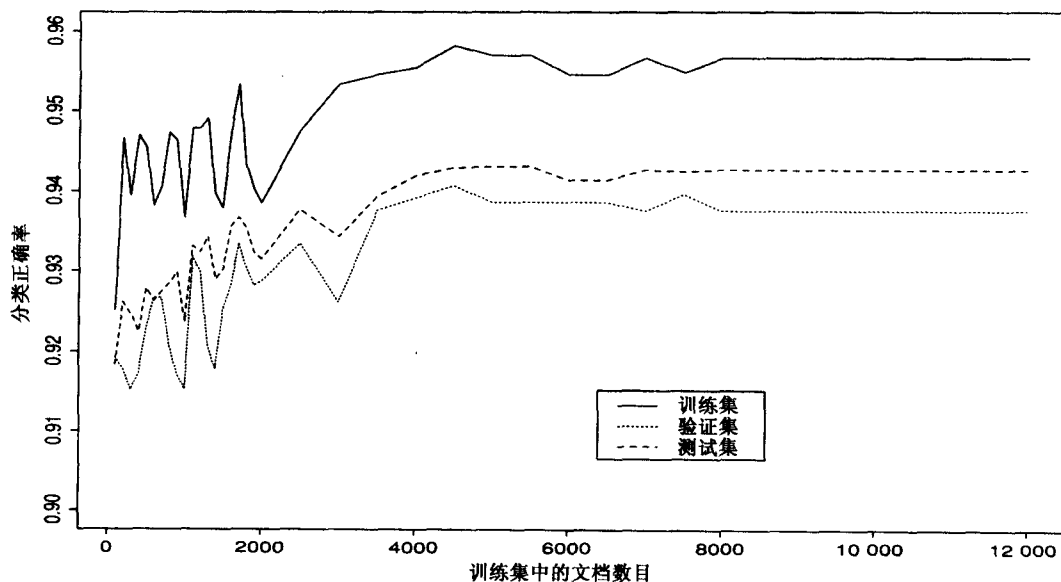


图 16.5 分类正确率和训练数据大小之间的关系图。其中 X 轴表示决策树训练集中包含的文档数目, Y 轴表示通过交叉验证法得出的决策树在测试集上的正确率。如图所示,分类正确率在小训练集下变化很大,随着训练数据的增加逐步提高并趋向一个稳定值

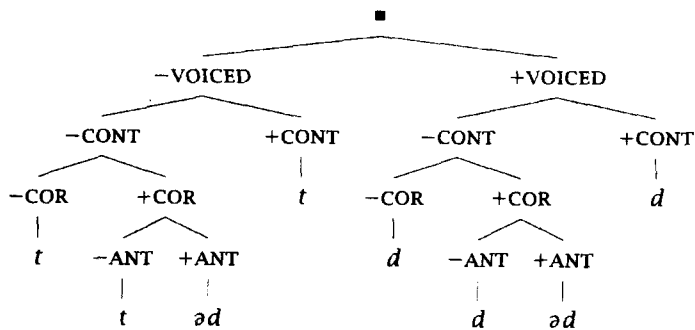


图 16.6 数据匮乏条件下利用决策树的音韵规则学习方法。除了一些特殊的规则外,如在 $[-\text{CONT}, +\text{COR}, +\text{ANT}]$ 后字符发 $/ə d/$ 音。通常清辅音后字符发 $/t/$ 音,浊音后字符发 $/d/$ 音,这是英语过去式中常见的规则。注意,发音特征具有最大的信息增益,所以决策树首先根据这个特征对训练数据进行划分,但同时剩余的特征很难再对数据分类,因为和它们相关的数据都已经被划分到不同的子树下

决策树的最大优点在于其结构清晰,易于理解。决策树分类过程非常清楚,从树根节点开始到树叶节点,整个分类过程很直观,利于跟踪。这一点不仅在调试代码和研究如何解决一个新领域的问题时有一些好处,更重要的是这种方法容易被外行人理解,这对开展合作研究以及在具体应用上都具备很大的价值。

16.2 最大熵建模

最大熵建模是将各种不同来源的信息的知识聚集到一个统一框架下。实际上所有分类问

题的问题空间都可以表述成特征形式。这些特征可能很复杂,但必须把和分类问题相关的先验知识描述出来。每个特征对应于模型中的一个限制条件,根据这些限制条件,选择一个满足最大熵条件的模型作为最后模型。这个原则表面上看起来和我们过去使用的原则相悖,因为本书很大篇幅都在讨论如何最小化模型对应数据的(交叉)熵的问题。但实际上,如果我们选择一个熵值相对较小的模型,其结果相当于人为地在建立的模型上增加一些信息限制条件,而导致这些限制条件的实验证据可能对我们来说是不可知的。最大熵的吸引人之处就在于熵值越大其不确定性越高。

本节中对特征选择的问题不多花费笔墨,它不是讨论的重点,前面提到的 20 维的特征在本节中继续使用。在最大熵建模过程中,特征选择和训练通常是结合在一起的。在理想的情况下,这种方法允许我们详细地列出所有潜在的与问题解决相关的信息,而把如何得到合适模型的各种参数的任务交给训练过程。本节只对最大熵的方法和思路做一些基本的介绍,较深层次的内容读者可以参照本章后面的深入阅读小节。

在特征本质上 f_i 是二值函数,用来刻画任何一个样本对 $(\vec{x}, c)$ 的任何属性,其中 $\vec{x}$ 是表示对象的向量(如表 16.3 中用一个 20 维的词条权重向量表示文档), c 是类别标记(1 表示该文档是 earnings 类别,0 表示其他类别)。针对文本分类问题,我们定义了以下的特征函数:

$$f_i(\vec{x}_j, c) = \begin{cases} 1 & \text{如果 } s_{ij} > 0 \text{ 且 } c = 1 \\ 0 & \text{其他} \end{cases} \quad (16.3)$$

其中 s_{ij} 表示路透社文档 j 中单词 i 的词条权重。注意,这里二值特征的使用和本章其他小节中的方法不同:其他的分类器算法使用的是权重值的大小,而不是用 0 和 1 这种记号来标记一个单词的存在与否^①。

给定特征集合后,首要的任务是基于训练集合计算每个特征的期望值,每个特征的限制条件都要求这个经验期望(empirical expectation)与模型中的理想特征期望值相同。在所有满足限制的分布模型中,选取满足使熵值最大化的分布。我们完全可以证明最大熵分布的惟一性,并存在一种广义比例迭代方法(generalized iterative scaling),确保迭代过程能够收敛。

最大熵模型存在着很多变种,本节重点介绍基于对数线性模型(loglinear model)的最大熵方法。

$$p(\vec{x}, c) = \frac{1}{Z} \prod_{i=1}^K \alpha_i^{f_i(\vec{x}, c)} \quad (16.4)$$

其中参数 K 表示参数的数目, α_i 为特征 f_i 的权重, Z 是归一化因数(通常称为“划分函数”),确保符合概率分布的形式。为了将此模型用于文本分类,需要计算 $p(\vec{x}, 0)$ 和 $p(\vec{x}, 1)$, 很简单,哪个概率值较大,就选取其为相应的类别标记。

值得注意的是,特征既包含了对象的类别信息,同时也包含了分类所需的度量信息。我们所介绍的是一种最广泛采用的特征定义方法,即忽略对象的所属类别,只引入对象的一些基本特征信息。

式(16.4)实际上定义了一个线性对数模型。如果我们对公式的两边做对数操作,那么

① 最大熵方法并没有限制特征必须为二值特征。下面将要介绍的广义比例迭代法就只需要权重非负而且其加权之和有界。但是,一般都使用二值特征,因为它能大大提高大规模重估计算的效率。

$\log p$ 就是几个对数函数的线性组合。

$$\log p(\vec{x}, c) = -\log Z + \sum_{i=1}^K f_i(\vec{x}, c) \log \alpha_i \quad (16.5)$$

在实际应用中,对数线性模型是解决分类问题的一种很重要也很常用的模型方法,其他一些常用的方法包括逻辑回归方法(McCullagh and Nelder 1989)、分解模型(decomposable model)(Bruce and Wiebe 1999)以及前面章节中提到的 HMM 和 PCFG 模型。之所以引入最大熵模型,是因为近年来它已经在统计自然语言处理中获得了广泛的应用,而且它也是最大熵理论的一个最重要的应用。

16.2.1 广义比例迭代

广义比例迭代方法本质上是寻找如公式(16.4)形式的最大熵分布 p^* , 并且满足以下的条件:

$$E_{p^*} f_i = E_{\tilde{p}} f_i \quad (16.6)$$

换句话说,要求分布 p^* 中特征 f_i 的期望值和训练集的经验分布相同,同时算法要求对所有样本 $(\vec{x}, c)$ 的特征和等于一个常数 C ^①:

$$\forall \vec{x}, c \quad \sum_i f_i(\vec{x}, c) = C \quad (16.7)$$

为了满足这个要求,我们定义 C 等于最大可能的特征和(对所有可能的数据,而不仅仅是观测数据):

$$C \triangleq \max_{\vec{x}, c} \sum_{i=1}^K f_i(\vec{x}, c)$$

同时增加一个额外特征 f_{K+1} , 定义如下:

$$f_{K+1}(\vec{x}, c) = C - \sum_{i=1}^K f_i(\vec{x}, c)$$

注意,相对于其他的特征定义,它不具备二值特征。

$E_p f_i$ 定义如下(见 2.1.5 节):

$$E_p f_i = \sum_{\vec{x}, c} p(\vec{x}, c) f_i(\vec{x}, c) \quad (16.8)$$

可以看出,公式中的求和是针对整个事件空间的,也就是说,针对所有可能的向量 $\vec{x}$ 和对应的类别标记 c 。而经验期望公式很容易用下式来表达:

$$E_{\tilde{p}} f_i = \sum_{\vec{x}, c} \tilde{p}(\vec{x}, c) f_i(\vec{x}, c) = \frac{1}{N} \sum_{j=1}^N f_i(\vec{x}_j, c) \quad (16.9)$$

其中 N 为训练集的大小。我们知道,对于任何一个样本,如果它不在训练数据中出现,那么它的经验分布概率为 0。

^① Berger et al. (1996)提出了一种改进的比例迭代算法(improved iterative scaling)。与广义比例迭代算法相比,它无需定义常数 C 。

实际上,最大熵分布 $E_{\mathbf{p}} f_i$ 很难直接计算出来,因为它要求遍历整个事件空间,对 $\vec{x}$ 和 c 的所有可能的组合求值,在本质上这是一个潜在的无限集,所以用下面只考虑能够观测到的 $\vec{x}$ 的公式来近似(Lau 1994:25)

$$E_{\mathbf{p}} f_i \approx \sum_{\vec{x}, c} \tilde{\mathbf{p}}(\vec{x}) \mathbf{p}(c|\vec{x}) f_i(\vec{x}, c) = \frac{1}{N} \sum_{j=1}^N \sum_c \mathbf{p}(c|\vec{x}_j) f_i(\vec{x}_j, c) \quad (16.10)$$

其中 c 遍历所有可能的类别,这里它取 0 或 1。

下面给出比例迭代算法的详细过程:

1. 初始化权重集合 $\{\alpha_i^{(1)}\}$, 可取任何值,但通常我们取所有的 $\alpha_i^{(1)} = 1, \forall 1 \leq j \leq K+1$, 计算 $E_{\tilde{\mathbf{p}}} f_i$, 取 $n = 1$ 。
2. 给定训练集中每一个样本 $(\vec{x}, c)$ 的 $\{\alpha_i^{(n)}\}$ 的取值后,对应于分布 $\mathbf{p}^{(n)}$ 计算 $\mathbf{p}^{(n)}(\vec{x}, c)$ 的值。

$$\mathbf{p}^{(n)}(\vec{x}, c) = \frac{1}{Z} \prod_{i=1}^{K+1} (\alpha_i^{(n)})^{f_i(\vec{x}, c)} \quad \text{其中} \quad Z = \sum_{\vec{x}, c} \prod_{i=1}^{K+1} (\alpha_i^{(n)})^{f_i(\vec{x}, c)} \quad (16.11)$$

3. 对于所有的 $\forall 1 \leq i \leq K+1$, 根据公式(16.10)计算 $E_{\tilde{\mathbf{p}}} f_i$ 。
4. 重新计算权重值 α_i

$$\alpha_i^{(n+1)} = \alpha_i^{(n)} \left(\frac{E_{\tilde{\mathbf{p}}} f_i}{E_{\mathbf{p}^{(n)}} f_i} \right)^{\frac{1}{c}} \quad (16.12)$$

5. 上述过程不断重复直到参数收敛,否则 n 值加 1,转到第二步。

为了方便阅读和理解,以上给出的是最大熵方法的简单易读的形式。然而,在实际的应用中,利用对数可能更容易计算。我们能够证明按照上面的算法可以收敛到满足限制(16.6)的分布 $\mathbf{p}^*$, 在所有满足这个限制的分布中,有且仅有一个满足最大熵条件的分布, Darroch and Ratcliff (1972)给出了存在性和惟一性的证明,简单的例子参见表 16.6。根据表 16.7 可做习题 16.9。

表 16.6 式(16.4)的形式是最大熵分布(maximum entropy distribution)的一个例子

$\vec{x}$	c	f_1	f_2	$\beta = f_1 \log \alpha_1 + f_2 \log \alpha_2$	2^β
profit	"earnings"				
(0)	0	0	1	1	2
(0)	1	0	1	1	2
(1)	0	0	1	1	2
(1)	1	1	0	2	4

在表 16.6 中,向量 $\vec{x}$ 值只包含单一元素,表明单词 profit 是否在文档出现。因为我们只定义了两种类别,属于 earning 类别或者不属于。特征 f_1 取 1,当且仅当文档属于 earning 类别, f_2 就是额外定义的 f_{K+1} 。我们定义权重值为 $\log \alpha_1 = 2.0, \log \alpha_2 = 1.0$, 归一化处理 ($Z = 2 + 2 + 2 + 4 = 10$), 在最大熵分布下可得出 $\mathbf{p}(0,0) = \mathbf{p}(0,1) = \mathbf{p}(1,0) = 2/Z = 0.2$, $\mathbf{p}(1,1) = 4/Z = 0.4$ 。有相同实验分布的一组数据的例子是 $((0,0), (0,1), (1,0), (1,1), (1,1))$ 。

表 16.7 对应于表 16.6 中最大熵分布的经验分布

profit 是否出现?	是不是 earnings 类?	
	YES	NO
YES	20	9
NO	8	13

16.2.2 文本分类的应用

我们在式(16.3)中给出了文本分类问题特征的定义,为了解决识别路透社中 earnings 类文档的类别判定问题,我们定义了 20 个特征,每一个对应于文档中筛选出的一个单词,另外在上一节的开始还引入了 f_{K+1} 特征,加入到原来的 20 个特征中。

我们在 9603 个训练集合文本上进行训练。表 16.8 给出了利用广义比例迭代算法计算出的权重值(500 次迭代),其中特征 *cts*, *profit*, *net* 和 *losst* 的权重值相对来说是最大的。如果把 $P(\text{"earnings"类}|\hat{x}) > P(\text{非 earnings 类}|\hat{x})$ 作为类别判断准则,表 16.9 就是我们最终的判断结果,其分类正确率为 96.2%。

表 16.8 为识别路透社 earnings 类别文档最大熵建模中的特征权重

词 w^i	特征 α_i	权重值 $\log_e \alpha_i$
vs	2.696	0.992
mln	1.079	0.076
cts	12.303	2.510
;	0.448	-0.803
&	0.450	-0.798
000	0.756	-0.280
loss	4.032	1.394
,	0.993	-0.007
"	1.502	0.407
3	0.435	-0.832
profit	9.701	2.272
dlrs	0.678	-0.388
1	1.193	0.177
pct	0.590	-0.528
is	0.418	-0.871
s	0.359	-1.025
that	0.703	-0.352
net	6.155	1.817
lt	3.566	1.271
at	0.490	-0.713
f_{K+1}	0.967	-0.034

表 16.9 对应表 16.8 的分布在测试集上的分类结果,分类正确率为 96.2%

分为 earnings 类别?	earnings 类别判断是否正确?	
	YES	NO
YES	1014	53
NO	73	2159

算法实现中也存在着不能回避的重要问题,即何时停止算法的迭代过程。其中一个可行的解决方法是计算经验的和估计的特征对数期望值的差($\log E_{\hat{p}} - \log E_p^{(n)}$),差值趋近于 0 可以作为算法终止的条件。Ristad(1996)提出了另一种替代方法,即通过在算法的迭代过程中观察什么时候趋于最大值来判断算法是否可以终止。如果这个最大的权重值太大,表明数据表示或者算法的实现本身存在问题。

本节中提到的最大熵框架什么时候能被应用到分类方法中去呢?现在实际应用于大规模数据的最大熵模型的一个特点是只能处理二值数据。在某些情况下这是最大熵模型的一个劣势。在文本分类中,我们经常需要利用到“证据力度”,而不仅是简单的证据出现与否。但是在这里这个缺点对我们似乎并没有什么影响。这也部分反映了这个分类问题有多么容易:简单的基于 *cts* 特征是不是零就可以获得 91.2% 的正确率。

此外,一个不可避免的问题是广义比例迭代算法的收敛速度可能很慢(Lau 1994)。对于二值分类问题,最大熵方法中的对数线性模型本身定义的线性分类面在原则上并不优于单纯贝叶斯或者线性回归方法,但很重要的一点是,除了分类算法本身的理论基础之外,训练的过程同样重要。不同于单纯贝叶斯方法,广义比例迭代方法考虑了特征之间的依存性。如果出现了同样的特征,那么特征的每个实例的权重都会减半。如果特征之间的依存性并不是问题,那么单纯贝叶斯方法是比最大熵模型更好的选择。

数据的平滑处理会带来一些问题。例如,如果一个特征对每个特定类别的判定具有决定性的作用,这个特征的权重值可能会过分得大。所以,我们需要在训练集中加入一些在现有训练集中并不存在的数据来实现平滑。在具体实现中,出现次数小于5的特征可以不予考虑。

最大熵方法的强项之一在于它提供了一个综合所有相关信息的框架。这是最大熵方法之所以吸引人的特点。它可以定义任意复杂的特征,只要实验者相信这些特征可以对类别的判定有所贡献。例如,Berger et al. (1996:57)建立了一个英法的机器翻译系统,其中针对两种语言中的介词 *pendant* 和 *in* 之间的互译问题定义了一个特征,即当且仅当 *in* 被翻译为 *pendant*,同时 *in* 后面的三个单词中含有单词 *weeks* 时,特征值被定义为1,否则为0。此外,在应用最大熵方法时我们也不用考虑特征的互异性问题和特征权重求值问题,而这两个问题在其他分类算法中往往又是较难处理的棘手问题。而且,最大熵模型完全基于最大熵原理:不额外加入任何在训练数据中没有观察到的信息。所以,在一定程度上最大熵建模为我们提供了一种有效整合不同信息源的概率模型框架。

这种方法的另一个强项是,它提供了一个特征选择和分类问题的整体框架。事实上,特征选择不能做到最大熵就是分类精度低的主要原因。绝大多数分类算法都不能很好地处理特征数量过大的问题。如果初始的特征维数很高,通常的方法是首先将特征集合的维数降到可操作的规模,正如我们这里所做的对词进行 X^2 测试。但是,在最大熵方法中我们可以利用所有潜在可能的特征集合,并扩展 Berger et al. (1996)所描述的基本方法,利用这个方法同时选择特征和适应模型。因为二者是整合的,所以可以对形成的分类模型给出一个清楚的概率解释(利用最大熵和极大似然),但这种解释对于其他方法如感知器和 k 近邻法就不那么清晰了。

本节只给出了最大熵方法的简单介绍,让读者有足够的信息去判断采用最大熵的方法解决面临的分类问题是否合适。如果想了解进一步的内容,可以参阅本章后面的深入阅读小节。

16.3 感知器

本节把感知器算法作为梯度下降(*gradient descent*)方法的一个简单例子加以介绍。在梯度下降方法中,我们的目的是最优化一个类似数据均方误差或似然率的准则函数。算法的每一步都计算函数的偏导函数,并且改变模型的参数使梯度下降速度最快。这种方法的出色之处在于梯度下降速度最快的方向正是我们寻找的算法性能可以提高的方向,当然这种所谓的“提高”是准则函数所定义的,比如说均方误差越小,算法的效果越好。

图 16.7 给出了详细的感知器学习算法。应用于文本分类问题时,我们采用和以前相同的文档表示方法,即文本向量的形式。感知器学习算法的目标是学习权重向量 $\vec{w}$ 和阈值 θ 的大小,当权重向量和文本向量的点积大于预定义的阈值时,就可以断定它所属的类别。具体在文本分类问题上(*earnings* 类文档被判断为 *yes*,非 *earnings* 类文档被判断为 *no*),分类函数定义如下:

文档类别被判断为 yes, 当且仅当 $\vec{w} \cdot \vec{x}_j = \sum_{i=1}^K w_i x_{ij} > \theta$

其中参数 K 为特征数(在我们的例子中 $K=20$), x_{ij} 为向量 $\vec{x}_j$ 的第 i 个成分。

感知器学习算法的基本思想很浅显易懂。如果依照当前的权重, 向量不能做出正确的判断, 算法会自动调整它(和 θ) 的值, 其调整方向依照事先定义的最优化准则函数, 即 $\sum_{i=1}^K w_i x_{ij} - \theta$ 。具体的例子读者可以参照图 16.8, 首先定义包含有参数 $\vec{w}$ 和 θ 的优化准则函数 ϕ ,

$$\phi(\vec{w}') = \phi\left(\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \\ \theta \end{pmatrix}\right) = \vec{w}' \cdot \vec{x}' = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_K \\ \theta \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \\ -1 \end{pmatrix}$$

这样 ϕ 的梯度算子(最大改变的方向)可以表示为向量 $\vec{x}'$

$$\nabla \phi(\vec{w}') = \vec{x}'$$

可以证明在所有可用 $\vec{x}$ 作为 $\vec{w}$ 的增量的向量集合中, $\vec{x}'$ 是 ϕ 变化量最大的一个, 它的方向代表了梯度递减的方向, 并且用在图 16.7 中作为迭代中的改变。

```

1  comment: Categorization Decision
2  funct decision( $\vec{x}$ ,  $\vec{w}$ ,  $\theta$ )  $\equiv$ 
3      if  $\vec{w} \cdot \vec{x} > \theta$  then
4          return yes
5      else
6          return no
7  fi.
9  comment: Initialization
10  $\vec{w} = 0$ 
11  $\theta = 0$ 
12 comment: Perceptron Learning Algorithm
13 while not converged yet do
14     for all elements  $\vec{x}_j$  in the training set do
15          $d = \text{decision}(\vec{x}_j, \vec{w}, \theta)$ 
16         if  $\text{class}(\vec{x}_j) = d$  then
17             continue
18         elseif  $\text{class}(\vec{x}_j) = \text{yes}$  and  $d = \text{no}$  then
19              $\theta = \theta - 1$ 
20              $\vec{w} = \vec{w} + \vec{x}_j$ 
21         elseif  $\text{class}(\vec{x}_j) = \text{no}$  and  $d = \text{yes}$  then
22              $\theta = \theta + 1$ 
23              $\vec{w} = \vec{w} - \vec{x}_j$ 
24         fi
25     end
26 end

```

图 16.7 感知器学习算法。感知器中 earnings 类别表示为 yes, 非 earnings 类别表示为 no。当权重向量和数据向量的点积大于阈值 θ 时, 算法判断文档类别为 yes, 否则判断类别为 no。感知器算法遍历了所有的样本实例, 如果当前权重向量对于某个实例判断正确, 那么它就保持不变, 否则就需要根据错误的方向在权重向量上加或者减去一个数值向量

图 16.8 给出的是针对二值分类问题的感知器算法的错误纠正步骤。同时,它也表明通过感知器学习的模式类别是线性可分的。权重向量相当于定义了一个决策超平面,在二值问题中它将向量空间分为正例和反例两个部分。图 16.8 中的决策面是 S ,它由权重 $\vec{w}$ 决定。如果分类问题中的类别可以被超平面划分,这样的分类问题称为线性可分(linearly separable)问题。

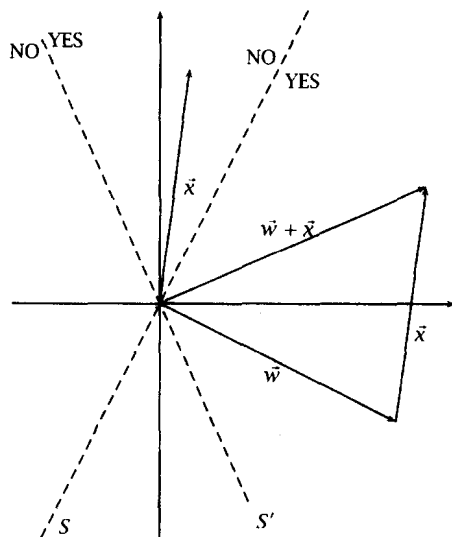


图 16.8 判断错误情况下感知器学习算法中的错误纠正过程。数据向量 $\vec{x}$ 被当前的权重向量 $\vec{w}$ 错判,从图中读者可以很明显地看出相对于分类界面 S ,它位于NO类的一侧。更正步骤为 $\vec{w}$ 加上 $\vec{x}$,改变了分类界面(图中表示为 S'), $\vec{x}$ 被 S' 正确地判断为YES类别

如果问题是线性可分的,存在严格的数学证明保证感知器学习算法的收敛性,我们称之为感知器收敛定理(perception convergence theorem)。从理论上讲,任何一个可分的问题都可以被感知器学习算法解决,因为可以根据被错误分类的元素调整权重向量,但是不排除同时也可能使原来被判断正确的数据在调整后反而被判错的情况。

表 16.10 中给出了迭代 1000 次之后,感知器学习算法学习到的判断 *earnings* 类别的权重。从中我们可以看出,和最大熵建模算法类似的是权重向量中对应单词 *cts*, *profit* 和 *lt* 等元素的权值最大。表 16.11 给出了在测试集合上的分类结果,正确率为 83.3%,这也表明这个问题不是线性可分的。相关内容参见后面的习题。

20 维的感知器显然很难直观地表示出来,为了理解方便,我们重新定义一个两维的感知器,包含 *mln* 和 *cts* 两个元素。图 16.9 中给出了权重向量和相应的决策面。

正如我们所知,感知器学习算法能确保解决线性可分的问题。当然,除了这里介绍的感知器算法之外,还存在很多其他的梯度下降法,但是其中大多数方法中最终收敛到的解向量都只能是局部最优解,而感知器学习算法的优点就在于它能收敛到全局最优解。但是,现实中有很多问题实际上是线性不可分的,很有名的一个问题就是所谓的异或问题,即假设存在两个特征 C_1 和 C_2 ,当 C_1 为真(True), C_2 为假(False)时,分类器判断为真,反之亦然。但是我们知道,决策树算法可以很简单地解决这个问题,但是感知器算法不可行。经过最初的对感知器学习方法的研究热潮(Rosenblatt, 1962)之后,大家都意识到了感知器算法的这个瓶颈,所以感知器的研究在这之后沉寂了十几年。Minsky and Papert (1969)可以看做对感知器研究的一个分界点,

此后感知器的相关研究逐渐被冷落。具体的介绍可以参见 Rumelhart and Zipser(1985),其中给出了感知器学习算法的研究发展史。

表 16.10 判断 earnings 类别的感知器,其中权重向量 $\vec{w}$ 和参数 θ 是基于路透社文档集由感知器学习算法学习得到的

词 w^i	权重
vs	11
mln	6
cts	24
;	2
&	12
000	-4
loss	19
'	-2
"	7
3	-7
profit	31
dtrs	1
1	3
pct	-4
is	-8
s	-12
that	-1
net	8
lt	11
at	-6
θ	37

表 16.11 表 16.10 中感知器的分类结果(路透社测试集),分类正确率为83.3%

分给 earnings 类?	分为 earnings 类正确吗?	
	YES	NO
YES	1059	521
NO	28	1691

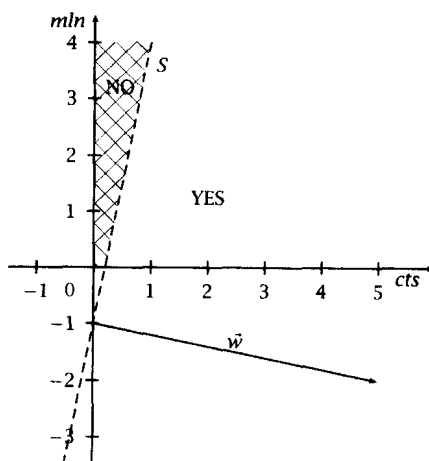


图 16.9 感知器学习算法的几何学解释。 cts 对应的权重为 5, mln 对应的权重为 -1, 我们把阈值定义为 1。图中线性分类界面 S 将第一象限(右上方)按照文档的类别分为 yes 和 no 两个区域。从中我们可以得出: 当文档中单词 mln 的出现次数大于 cts 时, 文档被判定为不属于 earnings 类别

当把感知器算法应用到非线性可分问题时,分界面有时会出现抖动情况,当算法寻找最优分类面失败时,算法的不稳定性体现得很明显。这也是文本分类中所遇到的问题。earnings文档的分类正确率也一直在 72% 和 93% 之间徘徊。我们的 83.3% 的结果是位于它们中间的一个结果,见表 16.10 和表 16.11。感知器学习算法还没有被广泛地应用到统计自然语言处理中,因为绝大多数的自然语言处理问题是非线性可分的。但是,如果一个问题线性可分的,那么应用感知器学习算法还是很有效的,因为它不仅形式简单,而且实现起来也不复杂。

在 20 世纪 80 年代,梯度下降算法再次成为研究的热点,这应该归功于一些新学习算法的提出,比如应用反向传播(backpropagation)训练多层感知器(multi-layer perceptrons, MLP),神经网络学习或者连接主义模型也都对梯度下降算法的研究有很大的作用。反向传播可以学习自然语言处理中的分类概念,包括类似异或等原来传统感知器学习算法不可解的问题,都可以应用反向传播算法得到解决。但是,它们也存在着一些难以克服的缺点,比如训练速度慢,结果是局部最优。

16.4 k 最近邻分类

最近邻分类原则(nearest neighbor classification rule)的基本原理非常简单:当判断一个新对象的类别时,从现有训练集中寻找一个最相似对象,根据这个最相似对象的类别属性来判断新对象的类别。

例如,在文本分类问题中,当训练集中有和当前文本完全等同的对象时(比如相同的向量表示),那么该对象的类别就是我们的判断结果,否则根据最相似文本来确定其类别。

我们把最近邻分类原则的一个推广形式称为 k 最近邻分类方法(KNN classification)。和只利用一个最近邻对象不同,在判断对象的类别时,我们可以参见多个而不是一个近似对象的模式类别属性。

KNN 算法的复杂之处在于如何确定对象之间的相似度。举例说明,假设一个问题的目标在于判断图像中是否包含有鹰的子图像。如果我们有一幅鹰的图像有待判别,数据库中存储了很多鹰的图片,但是按照 KNN 的方法可能做不出正确的判断,因为基于低层次图像特征做出的相似性判断是很不准确的。

幸运的是,在自然语言处理中定义了很多简单而且有效的相似度量。例如,在文本分类中,我们应用的是余弦相似法(见 8.5.1 节),选定 $k=1$,相应的“1NN”可以如下描述:

- 目标:给定训练数据集合 X ,对 $\vec{y}$ 进行分类。
- 在训练数据集合中,寻找和 $\vec{y}$ 具有最大相似度的训练样本。

$$\text{sim}_{\max}(\vec{y}) = \max_{\vec{x} \in X} \text{sim}(\vec{x}, \vec{y})$$
- 收集和 $\vec{y}$ 具有最大相似性的数据作为训练数据集合 X 的一个子集 A 。

$$A = \{\vec{x} \in X \mid \text{sim}(\vec{x}, \vec{y}) = \text{sim}_{\max}(\vec{y})\}$$
- 设 n_1 和 n_2 分别为集合 A 中类别 c_1 和 c_2 中的元素数目。根据这些数据计算样本的类别条件概率。

$$P(c_1|\vec{y}) = \frac{n_1}{n_1 + n_2} \qquad P(c_2|\vec{y}) = \frac{n_2}{n_1 + n_2} \qquad (16.13)$$

- 判断所属类别,当样本满足 $P(c_1|\hat{y}) > P(c_2|\hat{y})$ 时属于 c_1 , 否则样本属于 c_2 。

在路透社文档集合中,所有 3299 篇文档中有 2310 篇有一个最近邻的对象。在这 2310 篇文档中,有 697 篇文档和其最近邻对象的空间向量完全相同,其他 989 篇有多于 1 个的最近邻对象(其中有一篇文档有 247 个具备相同向量表达的最近邻文档)。当然这并不等于说,在路透社文档集合中有很多相同的文档,只是采用向量表示文档后的结果。

我们很容易将该算法推广到 k 大于 1 的情况,只需选择 k 个最近邻的对象,并对与不同类别近邻数相同的情况做出一些限制,然后根据多数原则做出最后的类别判定。在实际应用中,通常还可以按照相似度的不同赋予不同的权重,相似度越高,权重越大。

能够证明,对于足够大的训练数据集合,KNN 算法的错误率大约为贝叶斯算法的两倍。我们如果知道已知数据的真实分布,贝叶斯算法就是最优算法,其判断准则很简单(Duda and Hart 1973:98)。应用 1NN 算法解决 earnings 类文档分类问题的结果参见表 16.12。

表 16.12 1NN 算法对 earning 的文本
分类结果,正确率为95.3%

分为 earnings 类?	分为 earnings 类正确吗?	
	YES	NO
YES	1022	91
NO	65	2121

在上面的论述中,我们也间接提到 KNN 算法的缺陷,即依赖于正确定义的相似性量度。应用 KNN 算法解决问题时经常需要面对的一个问题就是相似度的调节。另一个潜在的问题是算法效率,计算一个样本和其他所有样本之间的相似度的花费要比线性分类算法或者决策树算法大得多。

但是,我们可以通过某种方法简化算法的计算,当相似性度量的定义非常明显时,KNN 算法是一个行之有效同时又相对简单的方法。

16.5 深入阅读

本章的目的在于为对自然语言处理中分类问题感兴趣的读者给出一些研究的方向。关于机器学习论述得较深入的最近一篇文献为 Mitchell(1997)。几种不同的机器学习方法应用于文本分类问题的效果可以参见 Yang(1999),Lewis et al.(1996)和 Schütze et al.(1995)。

本章中的特征以及基于这些特征的数据表示都可以从本书的配套网站上下载得到。

此外,由于篇幅所限,还有一些重要的分类算法在本章中没有涉及,比如逻辑回归法(Logistic Regression);线性判别式分析(Linear Discriminant Analysis)(Schütze et al. 1995);决策表(Decision List)。利用按顺序排列的规则列表分类(Yarowsky 1994);Winnow 算法,一种错误驱动的在线线性阈值学习算法(MistakeN-Driven Online Linear Threshold Learning Algorithm)(Dagan et al. 1997a);以及 Rocchio 算法(Rocchio 1971; Schapire et al. 1998)。

另外一种非常重要的分类算法,单纯贝叶斯算法,在 7.2.1 节中有相应的介绍,详细讨论参见 Domingos and Pazzani(1997),很出乎意料的是尽管它的独立性假设在大多数情况下都不成立,但是效果一般都很好。

利用决策树的自然语言处理任务包括句法分析(Magerman 1994)和词性标注(Schmid 1994)。Magerman 的做法是,在叶节点和根节点之间,利用留存数据训练一个差值模型,在确定

最优决策树大小时,不是利用传统的交叉验证方法,而是采用了另外的替代方法,训练多个决策树,然后平均化每个决策树的训练结果。这种方法就是我们经常提到的 Bagging 和 Boosting 算法,它们都是很有效的提高算法性能的方法(Breiman 1994; Quinlan 1996)。最早将决策树算法应用到文本分类中的尝试是 Lewis and Ringuette(1994)。

Jelinek(1997, ch. 13 ~ 14)给出了一个相当详尽的有关最大熵方法的论述,相关讨论也可以参见 Lau(1994)和 Ratnaparkhi(1997b)。Darroch and Ratcliff(1972)给出了最大熵方法的广义比例迭代方法,证明了它的收敛性质。Berger et al.(1996)和 Della Pietra et al.(1997)描述了特征选择的方法。

最大熵建模的应用包括词性标注(Ratnaparkhi, 1996)、文本分割(Reynar and Ratnaparkhi 1997)、介词短语附着(Ratnaparkhi 1998)、句子边界确定(Mikheev 1998)、指代消解(Kehler 1997)、专名识别(Borthwick et al. 1998)和句法分析(Skut and Brants 1998)。另外一个重要应用是语音识别中的语言建模问题(Lau et al. 1993; Rosenfeld 1994, 1996)。比例迭代拟合(Iterative proportional fitting)是另外一种与比例迭代算法相关的方法,最早被 Franz(1996, 1997)应用于词性标注和介词短语附着的对数线性模型的拟合问题。

在 20 世纪 80 年代至 20 世纪 90 年代,以 Rumelhart and McClelland(1986)以及 Elman(1990)为起点,神经网络算法或者多层感知器作为一种统计学上的方法在自然语言处理中重新得到重视。神经网络和反馈算法的介绍可以参见 Rumelhart et al.(1986), McClelland et al.(1986), Hertz et al.(1991)。其他和自然语言处理相关的神经网络的应用包括标注(Benello et al. 1989; Schütze 1993)、句子边界确定(Palmer and Hearst 1997)和句法分析(Henderson and Lane, 1998)。神经网络应用于文本分类的例子是(Wiener et al. 1995)和(Schütze et al. 1995)。Miikkulainen(1993)为自然语言处理任务搭建了一个通用神经网络模型。

图 16.7 中的感知器算法可以参见 Littlestone(1995)。其收敛性的证明见 Minsky and Papert(1988)以及 Duda and Hart(1973:142)中给出的介绍。

KNN 算法又称为基于记忆的算法(memory-based learning),在处理自然语言的问题时,应用得也很广泛,能够解决很多不同的问题。其中包括发音问题(Daelemans and van den Bosch 1996)、标注(Daelemans et al. 1996; Van Halteren et al. 1998)、介词短语附着(Zavrel et al. 1997)、浅层句法分析(Argamon et al. 1998)、词义消歧(Ng and Lee 1996)和平滑估计(Zavrel and Daelemans 1997)。KNN 解决文本分类问题的文献有(Yang 1994, 1995; Stanfill and Waltz 1986; Masand et al. 1992)和(Hull et al. 1996)。Yang(1994, 1995)建议了基于相似度大小对邻居加权的方法。一般来说,在 KNN 算法中我们采用余弦法作为度量相似度大小的方法(见 8.5.1 节),但是也可以采用欧几里得距离或差值法(Value Difference Metric)(Stanfill and Waltz 1986)。

16.6 习题

- 16.1 [★]当决策树只有单叶子节点时,earnings 类别的分类正确率是多少?
- 16.2 [★]在 7.1 节中,对于一个分类问题,当评估其分类器的分类效果时,我们定义了其分类的上界和下界概念,那么本章中 earnings 类别的上界和下界分别是什么?
- 16.3 [★★]文本分类问题的一个重要应用是垃圾邮件的检测,收集至少 100 条垃圾邮件和

非垃圾邮件,将它们分为测试集合和训练集合,构建一个决策树来检测垃圾邮件。设计特征是解决这类问题的关键,请认真设计你的特征集。

- 16.4 [★★]文本分类问题的另一个重要应用是“成人”内容的检测,即检测不适合未成年人阅读的与色情相关的内容。从网上下载一些相关的网页,训练决策树模型,建立一个网页过滤系统。
- 16.5 [★★]收集一些你自己和你的朋友所写的文章。为了增加训练集合的规模,我们可以将文本切割成一些小文本数据,训练一棵决策树来判断文章的作者。注意一些单词,例如 *because*, *though* 等词的使用频率,对判断作者有很好的启示作用。
- 16.6 [★★]从 Internet 网上下载一些英文文本和非英文文本,当然用其他的多语言资源也可以。建立一棵决策树区分这些文本的语言类别(见习题 6.10)。
- 16.7 [★]表 16.6 中的分布的类别决策函数是什么? 计算 $P(\text{"earnings"} | \text{profit})$ 和 $P(\text{"earnings"} | \text{非 profit})$ 的概率值?
- 16.8 [★]证明表 16.6 中的分布是广义比例迭代算法的一个迭代停止点,即只需迭代一次。
- 16.9 [★]计算表 16.7 中的分布的 E_p 值,证明它和表 16.6 中的分布的值相同。
- 16.10 [★]依照书中算法给出几步迭代计算,利用表 16.7 中给出的数据,说明算法可以收敛到表 16.6 中的分布形式(采用表 16.6 给出的特征定义)。
- 16.11 [★★]选取习题 16.3 到习题 16.6 中的任何一个作为原型,建立一个面向文本分类问题的最大熵模型。
- 16.12 [★★]针对一个具体的二值分类问题,编写一个可视化程序决策感知器,显示在训练和运行过程中分类界面的变化过程。
- 16.13 [★★]选择 *earnings* 和非 *earnings* 类别文档各 10 篇。确定两个单词作为分类特征,一个代表 X 轴,另一个代表 Y 轴,然后将这 20 篇训练文档加上类别标记。说明该分类问题是否是线性可分的?
- 16.14 [★]如何证明在两类问题中样本数据集合不是线性可分的?
- 16.15 [★]证明本章 *earnings* 类别文档集合的非线性可分性。
- 16.16 [★]假设某问题线性可分并训练一个感知器达到收敛,但是在这种情况下感知器的分类正确率经常不能达到 100%,为什么?
- 16.17 [★★]选择习题 16.3 到习题 16.6 中的任何一个,利用感知器的方法解决。
- 16.18 [★]本节中介绍的分类器算法中有两个算法具有线性决策边界,是哪两个呢?
- 16.19 [★]我们知道,当一个分类器的决策边界是线性的时,解决非线性问题的效果可能不是很好,是不是决策边界越复杂分类效果越好呢? 为什么不是这样? 参见 Roth(1998) 的讨论。
- 16.20 [★★]选择习题 16.3 到习题 16.6 中的任何一个,利用 k 最近邻算法解决文本分类问题。

附录 A 统计 表

下面给出的一些统计表并不能代替正规统计学书籍上的统计表以及相关的统计软件,但是相对于统计自然语言处理应用来说,这些统计表还是列出了一些关键的常用数据。

标准正态分布 (standard normal distribution)

下表给出了标准正态曲线下从 $-\infty$ 到特定 z 值区域的比例。

z	-3	-2	-1	0	1	2	3
比例	0.0013	0.023	0.159	0.5	0.841	0.977	0.9987

t 检验临界值 (t test critical value)

具有某个自由度的 t 分布由在 $-t^*$ 和 t^* 之间有百分比 C 的区域 (two-tailed), 以及曲线 t^* 到 ∞ 有 p 比例的区域 (onetailed) 组成。无穷自由度的值和 z 检验的临界值相同。

	p	0.05	0.025	0.01	0.005	0.001	0.0005
	C	90%	95%	98%	99%	99.8%	99.9%
d.f.	1	6.314	12.71	31.82	63.66	318.3	636.6
	10	1.812	2.228	2.764	3.169	4.144	4.587
	20	1.725	2.086	2.528	2.845	3.552	3.850
(z)	∞	1.645	1.960	2.326	2.576	3.091	3.291

χ^2 临界值

一个表项就是在一条特定自由度下,从 χ^{2*} 到 ∞ 的 χ^2 曲线上拥有 p 比例的区域点 χ^{2*} 。如果是一个 $r \times c$ 的表,就有 $(r-1)(c-1)$ 个自由度。

p	0.99	0.95	0.10	0.05	0.01	0.005	0.001	
d.f.	1	0.00016	0.0039	2.71	3.84	6.63	7.88	10.83
	2	0.020	0.10	4.60	5.99	9.21	10.60	13.82
	3	0.115	0.35	6.25	7.81	11.34	12.84	16.27
	4	0.297	0.71	7.78	9.49	13.28	14.86	18.47
	100	70.06	77.93	118.5	124.3	135.8	140.2	149.4

参考文献

The following conference abbreviations are used in this bibliography:

ACL *n* Proceedings of the *n*th Annual Meeting of the Association for Computational Linguistics

ANLP *n* Proceedings of the *n*th conference on Applied Natural Language Processing

COLING *n* Proceedings of the *n*th International Conference on Computational Linguistics (COLING-year)

EACL *n* Proceedings of the *n*th Conference of the European Chapter of the Association for Computational Linguistics

EMNLP *n* Proceedings of the *n*th Conference on Empirical Methods in Natural Language Processing

WVLC *n* Proceedings of the *n*th Workshop on Very Large Corpora

These conference proceedings are all available from the Association for Computational Linguistics, P.O. Box 6090, Somerset NJ 08875, USA, acl@aclweb.org, <http://www.aclweb.org>.

SIGIR 'y Proceedings of the (y - 77)th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval. Available from the Association for Computing Machinery, acmhelp@acm.org, <http://www.acm.org>.

Many papers are also available from the Computation and Language subject area of the Computing Research Repository e-print archive, a part of the xxx.lanl.gov e-print archive on the World Wide Web.

Abney, Steven. 1991. Parsing by chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny (eds.), *Principle-Based Parsing*, pp. 257-278. Dordrecht: Kluwer Academic.

Abney, Steven. 1996a. Part-of-speech tagging and partial parsing. In Steve Young and Gerrit Bloothoof (eds.), *Corpus-Based Methods in Language and Speech Processing*, pp. 118-136. Dordrecht: Kluwer Academic.

Abney, Steven. 1996b. Statistical methods and linguistics. In Judith L. Klavans and Philip Resnik (eds.), *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, pp. 1-26. Cambridge, MA: MIT Press.

Abney, Steven P. 1997. Stochastic attribute-value grammars. *Computational Linguistics* 23:597-618.

Ackley, D. H., G. E. Hinton, and T. J. Sejnowski. 1985. A learning algorithm for Boltzmann machines. *Cognitive Science* 9:147-169.

Aho, Alfred V., Ravi Sethi, and Jeffrey D. Ullman. 1986. *Compilers: Principles, Techniques, and Tools*. Reading, MA: Addison-Wesley.

- Allen, James. 1995. *Natural Language Understanding*. Redwood City, CA: Benjamin Cummings.
- Alshawi, Hiyan, Adam L. Buchsbaum, and Fei Xia. 1997. A comparison of head transducers and transfer for a limited domain translation application. In *ACL 35/EACL 8*, pp. 360-365.
- Alshawi, Hiyan, and David Carter. 1994. Training and scaling preference functions for disambiguation. *Computational Linguistics* 20:635-648.
- Anderson, John R. 1983. *The architecture of cognition*. Cambridge, MA: Harvard University Press.
- Anderson, John R. 1990. *The adaptive character of thought*. Hillsdale, NJ: Lawrence Erlbaum.
- Aone, Chinatsu, and Douglas McKee. 1995. Acquiring predicate-argument mapping information from multilingual texts. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 175-190. Cambridge, MA: MIT Press.
- Appelt, D. E., J. R. Hobbs, J. Bear, D. Israel, and M. Tyson. 1993. Fastus: A finite-state processor for information extraction from real-world text. In *Proc. of the 13th IJCAI*, pp. 1172-1178, Chambéry, France.
- Apresjan, Jurij D. 1974. Regular polysemy. *Linguistics* 142:5-32.
- Apté, Chidanand, Fred Damerau, and Sholom M. Weiss. 1994. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems* 12:233-251.
- Argamon, Shlomo, Ido Dagan, and Yuval Krymolowski. 1998. A memory-based approach to learning shallow natural language patterns. In *ACL 36/COLING 17*, pp. 67-73.
- Atwell, Eric. 1987. Constituent-likelihood grammar. In Roger Garside, Geoffrey Leech, and Geoffrey Sampson (eds.), *The Computational Analysis of English: A Corpus-Based Approach*. London: Longman.
- Baayen, Harald, and Richard Sproat. 1996. Estimating lexical priors for low-frequency morphologically ambiguous forms. *Computational Linguistics* 22: 155-166.
- Bahl, Lalit R., Frederick Jelinek, and Robert L. Mercer. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-5:179-190. Reprinted in (Waibel and Lee 1990), pp. 308-319.
- Bahl, Lalit R., and Robert L. Mercer. 1976. Part-of-speech assignment by a statistical decision algorithm. In *International Symposium on Information Theory*, Ronneby, Sweden.
- Baker, James K. 1975. Stochastic modeling for automatic speech understanding. In D. Raj Reddy (ed.), *Speech Recognition: Invited papers presented at the 1974 IEEE symposium*, pp. 521-541. New York: Academic Press. Reprinted in (Waibel and Lee 1990), pp. 297-307.

- Baker, James K. 1979. Trainable grammars for speech recognition. In D. H. Klatt and J. J. Wolf (eds.), *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pp. 547-550.
- Baldi, Pierre, and Søren Brunak. 1998. *Bioinformatics: The Machine Learning Approach*. Cambridge, MA: MIT Press.
- Barnbrook, Geoff. 1996. *Language and computers: a practical introduction to the computer analysis of language*. Edinburgh: Edinburgh University Press.
- Basili, Roberto, Maria Teresa Pazienza, and Paola Velardi. 1996. Integrating general-purpose and corpus-based verb classification. *Computational Linguistics* 22:559-568.
- Basili, Roberto, Gianluca De Rossi, and Maria Teresa Pazienza. 1997. Inducing terminology for lexical acquisition. In *EMNLP 2*, pp. 125-133.
- Baum, L. E., T. Petrie, G. Soules, and N. Weiss. 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics* 41:164-171.
- Beeferman, Doug, Adam Berger, and John Lafferty. 1997. Text segmentation using exponential models. In *EMNLP 2*, pp. 35-46.
- Bell, Timothy C., John G. Cleary, and Ian H. Witten. 1990. *Text Compression*. Englewood Cliffs, NJ: Prentice Hall.
- Benello, Julian, Andrew W. Mackie, and James A. Anderson. 1989. Syntactic category disambiguation with neural networks. *Computer Speech and Language* 3:203-217.
- Benson, Morton. 1989. The structure of the collocational dictionary. *International Journal of Lexicography* 2:1-14.
- Benson, Morton, Evelyn Benson, and Robert Ilson. 1993. *The BBI combinatory dictionary of English*. Amsterdam: John Benjamins.
- Berber Sardinha, A. P. 1997. *Automatic Identification of Segments in Written Texts*. PhD thesis, University of Liverpool.
- Berger, Adam L., Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics* 22:39-71.
- Berry, Michael W. 1992. Large-scale sparse singular value computations. *The International Journal of Supercomputer Applications* 6:13-49.
- Berry, Michael W., Susan T. Dumais, and Gavin W. O'Brien. 1995. Using linear algebra for intelligent information retrieval. *SIAM Review* 37:573-595.
- Berry, Michael W., and Paul G. Young. 1995. Using latent semantic indexing for multilanguage information retrieval. *Computers and the Humanities* 29: 413-429.
- Bever, Thomas G. 1970. The cognitive basis for linguistic structures. In J. R. Hayes (ed.), *Cognition and the development of language*. New York: Wiley.
- Biber, Douglas. 1993. Representativeness in corpus design. *Literary and Linguistic Computing* 8:243-257.

- Biber, Douglas, Susan Conrad, and Randi Reppen. 1998. *Corpus Linguistics: Investigating Language Structure and Use*. Cambridge: Cambridge University Press.
- Black, Ezra. 1988. An experiment in computational discrimination of English word senses. *IBM Journal of Research and Development* 32:185-194.
- Black, E., S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings, Speech and Natural Language Workshop*, pp. 306-311, Pacific Grove, CA. DARPA.
- Black, Ezra, Fred Jelinek, John Lafferty, David M. Magerman, Robert Mercer, and Salim Roukos. 1993. Towards history-based grammars: Using richer models for probabilistic parsing. In *ACL 31*, pp. 31-37. Also appears in the Proceedings of the DARPA Speech and Natural Language Workshop, Feb. 1992, pp. 134-139.
- Bod, Rens. 1995. *Enriching Linguistics with Statistics: Performance Models of Natural Language*. PhD thesis, University of Amsterdam.
- Bod, Rens. 1996. Data-oriented language processing: An overview. Technical Report LP-96-13, Institute for Logic, Language and Computation, University of Amsterdam.
- Bod, Rens. 1998. *Beyond Grammar: An experience-based theory of language*. Stanford, CA: CSLI Publications.
- Bod, Rens, and Ronald Kaplan. 1998. A probabilistic corpus-driven model for lexical-functional analysis. In *ACL 36/COLING 17*, pp. 145-151.
- Bod, Rens, Ron Kaplan, Remko Scha, and Khalil Sima'an. 1996. A data-oriented approach to lexical-functional grammar. In *Computational Linguistics in the Netherlands 1996*, Eindhoven, The Netherlands.
- Boguraev, Bran, and Ted Briscoe. 1989. *Computational Lexicography for Natural Language Processing*. London: Longman.
- Boguraev, Branimir, and James Pustejovsky. 1995. Issues in text-based lexicon acquisition. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 3-17. Cambridge MA: MIT Press.
- Boguraev, Branimir K. 1993. The contribution of computational lexicography. In Madeleine Bates and Ralph M. Weischedel (eds.), *Challenges in natural language processing*, pp. 99-132. Cambridge: Cambridge University Press.
- Bonnema, R. 1996. Data-oriented semantics. Master's thesis, Department of Computational Linguistics, University of Amsterdam.
- Bonnema, Remko, Rens Bod, and Remko Scha. 1997. A DOP model for semantic interpretation. In *ACL 35/EACL 8*, pp. 159-167.
- Bonzi, Susan, and Elizabeth D. Liddy. 1988. The use of anaphoric resolution for document description in information retrieval. In *SIGIR '88*, pp. 53-66.
- Bookstein, Abraham, and Don R. Swanson. 1975. A decision theoretic foundation for indexing. *Journal of the American Society for Information Science* 26:45-50.

- Booth, Taylor L. 1969. Probabilistic representation of formal languages. In *Tenth Annual IEEE Symposium on Switching and Automata Theory*, pp. 74-81.
- Booth, Taylor L., and Richard A. Thomson. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers* C-22:442-450.
- Borthwick, Andrew, John Sterling, Eugene Agichtein, and Ralph Grishman. 1998. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *WVLC 6*, pp. 152-160.
- Bourigault, Didier. 1993. An endogeneous corpus-based method for structural noun phrase disambiguation. In *EACL 6*, pp. 81-86.
- Box, George E. P., and George C. Tiao. 1973. *Bayesian Inference in Statistical Analysis*. Reading, MA: Addison-Wesley.
- Brants, Thorsten. 1998. Estimating Hidden Markov Model Topologies. In Jonathan Ginzburg, Zurab Khasidashvili, Carl Vogel, Jean-Jacques Lévy, and Enric Vallduví (eds.), *The Tbilisi Symposium on Logic, Language and Computation: Selected Papers*, pp. 163-176. Stanford, CA: CSLI Publications.
- Brants, Thorsten, and Wojciech Skut. 1998. Automation of treebank annotation. In *Proceedings of NeMLaP-98*, Sydney, Australia.
- Breiman, Leo. 1994. Bagging predictors. Technical Report 421, Department of Statistics, University of California at Berkeley.
- Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone. 1984. *Classification and Regression Trees*. Belmont, CA: Wadsworth International Group.
- Brent, Michael R. 1993. From grammar to lexicon: Unsupervised learning of lexical syntax. *Computational Linguistics* 19:243-262.
- Brew, Chris. 1995. Stochastic HPSG. In *EACL 7*, pp. 83-89.
- Brill, Eric. 1993a. Automatic grammar induction and parsing free text: A transformation-based approach. In *ACL 31*, pp. 259-265.
- Brill, Eric. 1993b. *A Corpus-Based Approach to Language Learning*. PhD thesis, University of Pennsylvania.
- Brill, Eric. 1993c. Transformation-based error-driven parsing. In *Proceedings Third International Workshop on Parsing Technologies*, Tilburg/Durbuy, The Netherlands/Belgium.
- Brill, Eric. 1995a. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics* 21:543-565.
- Brill, Eric. 1995b. Unsupervised learning of disambiguation rules for part of speech tagging. In *WVLC 3*, pp. 1-13.
- Brill, Eric, David Magerman, Mitch Marcus, and Beatrice Santorini. 1990. Deducing linguistic structure from the statistics of large corpora. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 275-282, San Mateo CA. Morgan Kaufmann.

- Brill, Eric, and Philip Resnik. 1994. A transformation-based approach to prepositional phrase attachment disambiguation. In *COLING 15*, pp. 1198-1204.
- Briscoe, Ted, and John Carroll. 1993. Generalized probabilistic LR parsing of natural language (corpora) with unification-based methods. *Computational Linguistics* 19:25-59.
- Britton, J. L. (ed.). 1992. *Collected Works of A. M. Turing: Pure Mathematics*. Amsterdam: North-Holland.
- Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics* 16: 79-85.
- Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, John D. Lafferty, and Robert L. Mercer. 1992a. Analysis, statistical transfer, and synthesis in machine translation. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*, pp. 83-100.
- Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer. 1992b. An estimate of an upper bound for the entropy of English. *Computational Linguistics* 18:31-40.
- Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1991a. A statistical approach to sense disambiguation in machine translation. In *Proceedings of the DARPA Workshop on Speech and Natural Language Workshop*, pp. 146-151.
- Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1991b. Word-sense disambiguation using statistical methods. In *ACL 29*, pp. 264-270.
- Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* 19:263-311.
- Brown, Peter F., Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992c. Class-based n -gram models of natural language. *Computational Linguistics* 18:467-479.
- Brown, Peter F., Jennifer C. Lai, and Robert L. Mercer. 1991c. Aligning sentences in parallel corpora. In *ACL 29*, pp. 169-176.
- Bruce, Rebecca, and Janyce Wiebe. 1994. Word-sense disambiguation using decomposable models. In *ACL 32*, pp. 139-145.
- Bruce, Rebecca F., and Janyce M. Wiebe. 1999. Decomposable modeling in natural language processing. *Computational Linguistics*. to appear.
- Brundage, Jennifer, Maren Kresse, Ulrike Schwall, and Angelika Storrer. 1992. Multiword lexemes: A monolingual and contrastive typology for natural language processing and machine translation. Technical Report 232, Institut fuer Wissensbasierte Systeme, IBM Deutschland GmbH, Heidelberg.
- Buckley, Chris, Amit Singhal, Mandar Mitra, and Gerard Salton. 1996. New retrieval approaches using SMART: TREC 4. In D. K. Harman (ed.), *The Second Text REtrieval Conference (TREC-2)*, pp. 25-48.

- Buitelaar, Paul. 1998. *CoreLex: Systematic Polysemy and Underspecification*. PhD thesis, Brandeis University.
- Burgess, Curt, and Kevin Lund. 1997. Modelling parsing constraints with high-dimensional context space. *Language and Cognitive Processes* 12:177-210.
- Burke, Robin, Kristian Hammond, Vladimir Kulyukin, Steven Lytinen, Noriko Tomuro, and Scott Schoenberg. 1997. Question answering from frequently asked question files. *AI Magazine* 18:57-66.
- Caraballo, Sharon A., and Eugene Charniak. 1998. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics* 24:275-298.
- Cardie, Claire. 1997. Empirical methods in information extraction. *AI Magazine* 18:65-79.
- Carletta, Jean. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics* 22:249-254.
- Carrasco, Rafael C., and Jose Oncina (eds.). 1994. *Grammatical inference and applications: second international colloquium, ICGI-94*. Berlin: Springer-Verlag.
- Carroll, G., and E. Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In Carl Weir, Stephen Abney, Ralph Grishman, and Ralph Weischedel (eds.), *Working Notes of the Workshop Statistically-Based NLP Techniques*, pp. 1-13. Menlo Park, CA: AAAI Press.
- Carroll, John. 1994. Relating complexity to practical performance in parsing with wide-coverage unification grammars. In *ACL* 32, pp. 287-294.
- Chang, Jason S., and Mathis H. Chen. 1997. An alignment method for noisy parallel corpora based on image processing techniques. In *ACL 35/EACL 8*, pp. 297-304.
- Chanod, Jean-Pierre, and Pasi Tapanainen. 1995. Tagging French - comparing a statistical and a constraint-based method. In *EACL* 7, pp. 149-156.
- Charniak, Eugene. 1993. *Statistical Language Learning*. Cambridge, MA: MIT Press.
- Charniak, Eugene. 1996. Tree-bank grammars. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI '96)*, pp. 1031-1036.
- Charniak, Eugene. 1997a. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI '97)*, pp. 598-603.
- Charniak, Eugene. 1997b. Statistical techniques for natural language parsing. *AI Magazine* pp. 33-43.
- Charniak, Eugene, Curtis Hendrickson, Neil Jacobson, and Mike Perkowitz. 1993. Equations for part-of-speech tagging. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pp. 784-789, Menlo Park, CA.
- Cheeseman, Peter, James Kelly, Matthew Self, John Stutz, Will Taylor, and Don Freeman. 1988. AutoClass: A Bayesian classification system. In *Proceedings of the Fifth International Conference on Machine Learning*, pp. 54-64, San Francisco, CA. Morgan Kaufmann.

- Chelba, Ciprian, and Frederick Jelinek. 1998. Exploiting syntactic structure for language modeling. In *ACL 36/COLING 17*, pp. 225-231.
- Chen, Jen Nan, and Jason S. Chang. 1998. Topical clustering of MRD senses based on information retrieval techniques. *Computational Linguistics* 24:61-95.
- Chen, Stanley F. 1993. Aligning sentences in bilingual corpora using lexical information. In *ACL 31*, pp. 9-16.
- Chen, Stanley F. 1995. Bayesian grammar induction for language modeling. In *ACL 33*, pp. 228-235.
- Chen, Stanley F., and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *ACL 34*, pp. 310-318.
- Chen, Stanley F., and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University.
- Chi, Zhiyi, and Stuart Geman. 1998. Estimation of probabilistic context-free grammars. *Computational linguistics* 24:299-305.
- Chitrao, Mahesh V., and Ralph Grishman. 1990. Statistical parsing of messages. In *Proceedings of the DARPA Speech and Natural Language Workshop, Hidden Valley, PA*, pp. 263-266. Morgan Kaufmann.
- Chomsky, Noam. 1957. *Syntactic Structures*. The Hague: Mouton.
- Chomsky, Noam. 1965. *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.
- Chomsky, Noam. 1980. *Rules and Representations*. New York: Columbia University Press.
- Chomsky, Noam. 1986. *Knowledge of Language: Its Nature, Origin, and Use*. New York: Prager.
- Chomsky, Noam. 1995. *The Minimalist Program*. Cambridge, MA: MIT Press.
- Choueika, Yaacov. 1988. Looking for needles in a haystack or locating interesting collocational expressions in large textual databases. In *Proceedings of the RIAO*, pp. 43-38.
- Choueika, Yaacov, and Serge Lúsignan. 1985. Disambiguation by short contexts. *Computers and the Humanities* 19:147-158.
- Church, Kenneth, William Gale, Patrick Hanks, and Donald Hindle. 1991. Using statistics in lexical analysis. In Uri Zernik (ed.), *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 115-164. Hillsdale, NJ: Lawrence Erlbaum.
- Church, Kenneth, and Ramesh Patil. 1982. Coping with syntactic ambiguity or how to put the block in the box on the table. *Computational Linguistics* 8: 139-149.
- Church, Kenneth W. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *ANLP 2*, pp. 136-143.
- Church, Kenneth Ward. 1993. Char_align: A program for aligning parallel texts at the character level. In *ACL 31*, pp. 1-8.
- Church, Kenneth Ward. 1995. One term or two? In *SIGIR '95*, pp. 310-318.

- Church, Kenneth W., and William A. Gale. 1991a. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language* 5:19-54.
- Church, Kenneth W., and William A. Gale. 1991b. Concordances for parallel text. In *Proceedings of the Seventh Annual Conference of the UW Centre for the New OED and Text Research*, pp. 40-62, Oxford.
- Church, Kenneth W., and William A. Gale. 1995. Poisson mixtures. *Natural Language Engineering* 1:163-190.
- Church, Kenneth Ward, and Patrick Hanks. 1989. Word association norms, mutual information and lexicography. In *ACL* 27, pp. 76-83.
- Church, Kenneth Ward, and Mark Y. Liberman. 1991. A status report on the ACL/DCI. In *Proceedings of the 7th Annual Conference of the UW Centre for New OED and Text Research: Using Corpora*, pp. 84-91.
- Church, Kenneth W., and Robert L. Mercer. 1993. Introduction to the special issue on computational linguistics using large corpora. *Computational Linguistics* 19:1-24.
- Clark, Eve, and Herbert Clark. 1979. When nouns surface as verbs. *Language* 55: 767-811.
- Cleverdon, Cyril W., and J. Mills. 1963. The testing of index language devices. *Aslib Proceedings* 15:106-130. Reprinted in (Sparck Jones and Willett 1998).
- Coates-Stephens, Sam. 1993. The analysis and acquisition of proper names for the understanding of free text. *Computers and the Humanities* 26:441-456.
- Collins, Michael John. 1996. A new statistical parser based on bigram lexical dependencies. In *ACL* 34, pp. 184-191.
- Collins, Michael John. 1997. Three generative, lexicalised models for statistical parsing. In *ACL 35/EACL* 8, pp. 16-23.
- Collins, Michael John, and James Brooks. 1995. Prepositional phrase attachment through a backed-off model. In *WVLC* 3, pp. 27-38.
- Copestake, Ann, and Ted Briscoe. 1995. Semi-productive polysemy and sense extension. *Journal of Semantics* 12:15-68.
- Cormen, Thomas H., Charles E. Leiserson, and Ronald L. Rivest. 1990. *Introduction to Algorithms*. Cambridge, MA: MIT Press.
- Cottrell, Garrison W. 1989. *A Connectionist Approach to Word Sense Disambiguation*. London: Pitman.
- Cover, Thomas M., and Joy A. Thomas. 1991. *Elements of Information Theory*. New York: John Wiley & Sons.
- Cowart, Wayne. 1997. *Experimental syntax: Applying objective methods to sentence judgments*. Thousand Oaks, CA: Sage Publications.
- Croft, W. B., and D. J. Harper. 1979. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation* 35:285-295.
- Crowley, Terry, John Lynch, Jeff Siegel, and Julie Piau. 1995. *The Design of Language: An introduction to descriptive linguistics*. Auckland: Longman Paul.

- Crystal, David. 1987. *The Cambridge Encyclopedia of Language*. Cambridge, England: Cambridge University Press.
- Cutting, Doug, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1991. A practical part-of-speech tagger. In *ANLP 3*, pp. 133-140.
- Cutting, Douglas R., David R. Karger, and Jan O. Pedersen. 1993. Constant interaction-time scatter/gather browsing of very large document collections. In *SIGIR '93*, pp. 126-134.
- Cutting, Douglas R., Jan O. Pedersen, David Karger, and John W. Tukey. 1992. Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR '92*, pp. 318-329.
- Daelemans, Walter, and Antal van den Bosch. 1996. Language-independent data-oriented grapheme-to-phoneme conversion. In J. Van Santen, R. Sproat, J. Olive, and J. Hirschberg (eds.), *Progress in Speech Synthesis*, pp. 77-90. New York: Springer Verlag.
- Daelemans, Walter, Jakub Zavrel, Peter Berck, and Steven Gillis. 1996. MBT: A memory-based part of speech tagger generator. In *WVLC 4*, pp. 14-27.
- Dagan, Ido, Kenneth Church, and William Gale. 1993. Robust bilingual word alignment for machine aided translation. In *WVLC 1*, pp. 1-8.
- Dagan, Ido, and Alon Itai. 1994. Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics* 20:563-596.
- Dagan, Ido, Alon Itai, and Ulrike Schwall. 1991. Two languages are more informative than one. In *ACL 29*, pp. 130-137.
- Dagan, Ido, Yael Karov, and Dan Roth. 1997a. Mistake-driven learning in text categorization. In *EMNLP 2*, pp. 55-63.
- Dagan, Ido, Lillian Lee, and Fernando Pereira. 1997b. Similarity-based methods for word sense disambiguation. In *ACL 35/EACL 8*, pp. 56-63.
- Dagan, Ido, Fernando Pereira, and Lillian Lee. 1994. Similarity-based estimation of word cooccurrence probabilities. In *ACL 32*, pp. 272-278.
- Damerau, Fred J. 1993. Generating and evaluating domain-oriented multi-word terms from texts. *Information Processing & Management* 29:433-447.
- Darroch, J. N., and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics* 43:1470-1480.
- de Saussure, Ferdinand. 1962. *Cours de linguistique générale*. Paris: Payot.
- Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41:391-407.
- DeGroot, Morris H. 1975. *Probability and Statistics*. Reading, MA: Addison-Wesley.
- Della Pietra, Stephen, Vincent Della Pietra, and John Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19.

- Demers, A.J. 1977. Generalized left corner parsing. In *Proceedings of the Fourth Annual ACM Symposium on Principles of Programming Languages*, pp. 170-181.
- Dempster, A.P., N.M. Laird, and D.B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society Series B* 39: 1-38.
- Dermatas, Evangelos, and George Kokkinakis. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics* 21:137-164.
- DeRose, Steven J. 1988. Grammatical category disambiguation by statistical optimization. *Computational Linguistics* 14:31-39.
- Deroquault, Anne-Marie, and Bernard Merialdo. 1986. Natural language modeling for phoneme-to-text transcription. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8:742-649.
- Dietterich, Thomas G. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation* 10:1895-1924.
- Dini, Luca, Vittorio Di Tomaso, and Frédérique Segond. 1998. Error-driven word sense disambiguation. In *ACL 36/COLING 17*, pp. 320-324.
- Dolan, William B. 1994. Word sense ambiguity: Clustering related senses. In *COLING 15*, pp. 712-716.
- Dolin, Ron. 1998. *Pharos: A Scalable Distributed Architecture for Locating Heterogeneous Information Sources*. PhD thesis, University of California at Santa Barbara.
- Domingos, Pedro, and Michael Pazzani. 1997. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning* 29:103-130.
- Doran, Christy, Dania Egedi, Beth Ann Hockey, B. Srinivas, and Martin Zaidel. 1994. XTAG system - a wide coverage grammar for English. In *COLING 15*, pp. 922-928.
- Dorr, Bonnie J., and Mari Broman Olsen. 1997. Deriving verbal and compositional lexical aspect for nlp applications. In *ACL 35/EACL 8*, pp. 151-158.
- Dras, Mark, and Mike Johnson. 1996. Death and lightness: Using a demographic model to find support verbs. In *Proceedings of the 5th International Conference on the Cognitive Science of Natural Language Processing*, Dublin.
- Duda, Richard O., and Peter E. Hart. 1973. *Pattern classification and scene analysis*. New York: Wiley.
- Dumais, Susan T. 1995. Latent semantic indexing (LSI): TREC-3 report. In *The Third Text REtrieval Conference (TREC 3)*, pp. 219-230.
- Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics* 19:61-74.
- Dunning, Ted. 1994. Statistical identification of language. Technical report, Computing Research Laboratory, New Mexico State University.

- Durbin, Richard, Sean Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge: Cambridge University Press.
- Eeg-Olofsson, Mats. 1985. A probability model for computer-aided word class determination. *Literary and Linguistic Computing* 5:25-30.
- Egan, Dennis E., Joel R. Remde, Louis M. Gomez, Thomas K. Landauer, Jennifer Eberhardt, and Carol C. Lochbaum. 1989. Formative design-evaluation of superbook. *ACM Transactions on Information Systems* 7:30-57.
- Eisner, Jason. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 16*, pp. 340-345.
- Ellis, C. A. 1969. *Probabilistic Languages and Automata*. PhD thesis, University of Illinois. Report No. 355, Department of Computer Science.
- Elman, Jeffrey L. 1990. Finding structure in time. *Cognitive Science* 14:179-211.
- Elworthy, David. 1994. Does Baum-Welch re-estimation help taggers? In *ANLP 4*, pp. 53-58.
- Estoup, J. B. 1916. *Gammes Sténographiques*, 4th edition. Paris.
- Evans, David A., Kimberly Ginther-Webster, Mary Hart, Robert G. Lefferts, and Ira A. Monarch. 1991. Automatic indexing using selective NLP and first-order thesauri. In *Proceedings of the RIAO*, volume 2, pp. 624-643.
- Evans, David A., and Chengxiang Zhai. 1996. Noun-phrase analysis in unrestricted text for information retrieval. In *ACL 34*, pp. 17-24.
- Fagan, Joel L. 1987. Automatic phrase indexing for document retrieval: An examination of syntactic and non-syntactic methods. In *SIGIR '87*, pp. 91-101.
- Fagan, Joel L. 1989. The effectiveness of a nonsyntactic approach to automatic phrase indexing for document retrieval. *Journal of the American Society for Information Science* 40:115-132.
- Fano, Robert M. 1961. *Transmission of information; a statistical theory of communications*. New York: MIT Press.
- Fillmore, Charles J., and B. T. S. Atkins. 1994. Starting where the dictionaries stop: The challenge of corpus lexicography. In B.T.S. Atkins and A. Zampolli (eds.), *Computational Approaches to the Lexicon*, pp. 349-393. Oxford: Oxford University Press.
- Finch, Steven, and Nick Chater. 1994. Distributional bootstrapping: From word class to proto-sentence. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, pp. 301-306, Hillsdale, NJ. Lawrence Erlbaum.
- Finch, Steven Paul. 1993. *Finding Structure in Language*. PhD thesis, University of Edinburgh.
- Firth, J. R. 1957. A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis*, pp. 1-32. Oxford: Philological Society. Reprinted in F. R. Palmer (ed), *Selected Papers of J. R. Firth 1952-1959*, London: Longman, 1968.
- Fisher, R. A. 1922. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society* 222:309-368.

- Fontenelle, Thierry, Walter Brls, Luc Thomas, Tom Vanallemeersch, and Jacques Jansen. 1994. DECIDE, MLAP-Project 93-19, deliverable D-1a: survey of collocation extraction tools. Technical report, University of Liege, Liege, Belgium.
- Ford, Marilyn, Joan Bresnan, and Ronald M. Kaplan. 1982. A competence-based theory of syntactic closure. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*, pp. 727-796. Cambridge, MA: MIT Press.
- Foster, G. F. 1991. Statistical lexical disambiguation. Master's thesis, School of Computer Science, McGill University.
- Frakes, William B., and Ricardo Baeza-Yates (eds.). 1992. *Information Retrieval*. Englewood Cliffs, NJ: Prentice Hall.
- Francis, W. Nelson, and Henry Kuera. 1964. *Manual of information to accompany a standard corpus of present-day edited American English, for use with digital computers*. Providence, RI: Dept of Linguistics, Brown University.
- Francis, W. Nelson, and Henry Kuera. 1982. *Frequency Analysis of English Usage: Lexicon and Grammar*. Boston, MA: Houghton Mifflin.
- Franz, Alexander. 1996. *Automatic Ambiguity Resolution in Natural Language Processing*, volume 1171 of *Lecture Notes in Artificial Intelligence*. Berlin: Springer Verlag.
- Franz, Alexander. 1997. Independence assumptions considered harmful. In *ACL 35/EACL 8*, pp. 182-189.
- Franz, Alexander Mark. 1995. *A Statistical Approach to Syntactic Ambiguity Resolution*. PhD thesis, CMU.
- Frazier, Lyn. 1978. *On Comprehending Sentences: Syntactic Parsing Strategies*. PhD thesis, University of Connecticut.
- Freedman, David, Robert Pisani, and Roger Purves. 1998. *Statistics*. New York: W. W. Norton. 3rd ed.
- Friedl, Jeffrey E. F. 1997. *Mastering Regular Expressions*. Sebastopol, CA: O'Reilly & Associates.
- Fu, King-Sun. 1974. *Syntactic Methods in Pattern Recognition*. London: Academic Press.
- Fung, Pascale, and Kenneth W. Church. 1994. K-vec: A new approach for aligning parallel texts. In *COLING 15*, pp. 1096-1102.
- Fung, Pascale, and Kathleen McKeown. 1994. Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic time warping. In *Proceedings of the Association for Machine Translation in the Americas (AMTA-94)*, pp. 81-88.
- Gale, William A., and Kenneth W. Church. 1990a. Estimation procedures for language context: Poor estimates of context are worse than none. In *Proceedings in Computational Statistics (COMPSTAT 9)*, pp. 69-74.
- Gale, William A., and Kenneth W. Church. 1990b. Poor estimates of context are worse than none. In *Proceedings of the June 1990 DARPA Speech and Natural Language Workshop*, pp. 283-287, Hidden Valley, PA.
- Gale, William A., and Kenneth W. Church. 1991. A program for aligning sentences in bilingual corpora. In *ACL 29*, pp. 177-184.

- Gale, William A., and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics* 19:75-102.
- Gale, William A., and Kenneth W. Church. 1994. What's wrong with adding one? In Nelleke Oostdijk and Pieter de Haan (eds.), *Corpus-Based Research into Language: in honour of Jan Aarts*. Amsterdam: Rodopi.
- Gale, William A., Kenneth W. Church, and David Yarowsky. 1992a. Estimating upper and lower bounds on the performance of word-sense disambiguation programs. In *ACL 30*, pp. 249-256.
- Gale, William A., Kenneth W. Church, and David Yarowsky. 1992b. A method for disambiguating word senses in a large corpus. *Computers and the Humanities* 26:415-439.
- Gale, William A., Kenneth W. Church, and David Yarowsky. 1992c. A method for disambiguating word senses in a large corpus. Technical report, AT&T Bell Laboratories, Murray Hill, NJ.
- Gale, William A., Kenneth W. Church, and David Yarowsky. 1992d. Using bilingual materials to develop word sense disambiguation methods. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, pp. 101-112.
- Gale, William A., Kenneth W. Church, and David Yarowsky. 1992e. Work on statistical methods for word sense disambiguation. In Robert Goldman, Peter Norvig, Eugene Charniak, and Bill Gale (eds.), *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pp. 54-60, Menlo Park, CA. AAAI Press.
- Gale, William A., and Geoffrey Sampson. 1995. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics* 2:217-237.
- Gallager, Robert G. 1968. *Information theory and reliable communication*. New York: Wiley.
- Garside, Roger. 1995. Grammatical tagging of the spoken part of the British National Corpus: a progress report. In Geoffrey N. Leech, Greg Myers, and Jenny Thomas (eds.), *Spoken English on computer: transcription, mark-up, and application*. Harlow, Essex: Longman.
- Garside, Roger, and Fanny Leech. 1987. The UCREL probabilistic parsing system. In Roger Garside, Geoffrey Leech, and Geoffrey Sampson (eds.), *The Computational Analysis of English: A Corpus-Based Approach*, pp. 66-81. London: Longman.
- Garside, Roger, Geoffrey Sampson, and Geoffrey Leech (eds.). 1987. *The Computational analysis of English: a corpus-based approach*. London: Longman.
- Gaussier, Éric. 1998. Flow network models for word alignment and terminology extraction from bilingual corpora. In *ACL 36/COLING 17*, pp. 444-450.
- Ge, Niyu, John Hale, and Eugene Charniak. 1998. A statistical approach to anaphora resolution. In *WVLC 6*, pp. 161-170.
- Ghahramani, Zoubin. 1994. Solving inverse problems using an EM approach to density estimation. In Michael C. Mozer, Paul Smolensky, David S. Touretzky, and Andreas S. Weigend (eds.), *Proceedings of the 1993 Connectionist Models Summer School*, Hillsdale, NJ. Erlbaum Associates.

- Gibson, Edward, and Neal J. Pearlmutter. 1994. A corpus-based analysis of psycholinguistic constraints on prepositional-phrase attachment. In Charles Clifton, Jr., Lyn Frazier, and Keith Rayner (eds.), *Perspectives on Sentence Processing*, pp. 181-198. Hillsdale, NJ: Lawrence Erlbaum.
- Gold, E. Mark. 1967. Language identification in the limit. *Information and Control* 10:447-474.
- Goldszmidt, Moises, and Mehran Sahami. 1998. A probabilistic approach to full-text document clustering. Technical Report SIDL-WP-1998-0091, Stanford Digital Library Project, Stanford, CA.
- Golub, Gene H., and Charles F. van Loan. 1989. *Matrix Computations*. Baltimore: The Johns Hopkins University Press.
- Good, I. J. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika* 40:237-264.
- Good, I. J. 1979. Studies in the history of probability and statistics. XXXVII: A. M. Turing's statistical work in World War II. *Biometrika* 66:393-396.
- Goodman, Joshua. 1996. Parsing algorithms and metrics. In *ACL 34*, pp. 177-183.
- Greenbaum, Sidney. 1993. The tagset for the International Corpus of English. In Eric Atwell and Clive Souter (eds.), *Corpus-based Computational Linguistics*, pp. 11-24. Amsterdam: Rodopi.
- Greene, Barbara B., and Gerald M. Rubin. 1971. Automatic grammatical tagging of English. Technical report, Brown University, Providence, RI.
- Grefenstette, Gregory. 1992a. Finding semantic similarity in raw text: the deese antonyms. In Robert Goldman, Peter Norvig, Eugene Charniak, and Bill Gale (eds.), *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pp. 61-65, Menlo Park, CA. AAAI Press.
- Grefenstette, Gregory. 1992b. Use of syntactic context to produce term association lists for text retrieval. In *SIGIR '92*, pp. 89-97.
- Grefenstette, Gregory. 1994. *Explorations in Automatic Thesaurus Discovery*. Boston: Kluwer Academic Press.
- Grefenstette, Gregory. 1996. Evaluation techniques for automatic semantic extraction: Comparing syntactic and window-based approaches. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, chapter 11, pp. 205-216. Cambridge, MA: MIT Press.
- Grefenstette, Gregory (ed.). 1998. *Cross-language information retrieval*. Boston, MA: Kluwer Academic Publishers.
- Grefenstette, Gregory, and Pasi Tapanainen. 1994. What is a word, what is a sentence? Problems of tokenization. In *Proceedings of the Third International Conference on Computational Lexicography (COMPLEX '94)*, pp. 79-87, Budapest. Available as Rank Xerox Research Centre technical report MLTT-004.
- Grenander, Ulf. 1967. Syntax-controlled probabilities. Technical report, Division of Applied Mathematics, Brown University.

- Günter, R., L. B. Levitin, B. Shapiro, and P. Wagner. 1996. Zipf's law and the effect of ranking on probability distributions. *International Journal of Theoretical Physics* 35:395-417.
- Guthrie, Joe A., Louise Guthrie, Yorick Wilks, and Homa Aidinejad. 1991. Subject-dependent co-occurrence and word sense disambiguation. In *ACL 29*, pp. 146-152.
- Guthrie, Louise, James Pustejovsky, Yorick Wilks, and Brian M. Slator. 1996. The role of lexicons in natural language processing. *Communications of the ACM* 39:63-72.
- Halliday, M. A. K. 1966. Lexis as a linguistic level. In C. E. Bazell, J. C. Catford, M. A. K. Halliday, and R. H. Robins (eds.), *In memory of J. R. Firth*, pp. 148-162. London: Longmans.
- Halliday, M. A. K. 1994. *An introduction to functional grammar*, 2nd edition. London: Edward Arnold.
- Harman, D.K. (ed.). 1996. *The Third Text REtrieval Conference (TREC-4)*. Washington DC: U.S. Department of Commerce.
- Harman, D. K. (ed.). 1994. *The Second Text REtrieval Conference (TREC-2)*. Washington DC: U.S. Department of Commerce. NIST Special Publication 500-215.
- Harnad, Stevan (ed.). 1987. *Categorical perception: the groundwork of cognition*. Cambridge: Cambridge University Press.
- Harris, B. 1988. Bi-text, a new concept in translation theory. *Language Monthly* 54.
- Harris, T. E. 1963. *The Theory of Branching Processes*. Berlin: Springer.
- Harris, Zellig. 1951. *Methods in Structural Linguistics*. Chicago: University of Chicago Press.
- Harrison, Philip, Steven Abney, Ezra Black, Dan Flickinger, Claudia Gdaniec, Ralph Grishman, Donald Hindle, Robert Ingria, Mitch Marcus, Beatrice Santorini, and Tomek Strzalkowski. 1991. Natural Language Processing Systems Evaluation Workshop, Technical Report RL-TR-91-362. In Jeannette G. Neal and Sharon M. Walter (eds.), *Evaluating Syntax Performance of Parser/Grammars of English*, Rome Laboratory, Air Force Systems Command, Griffis Air Force Base, NY 13441-5700.
- Harter, Steve. 1975. A probabilistic approach to automatic keyword indexing: Part II. an algorithm for probabilistic indexing. *Journal of the American Society for Information Science* 26:280-289.
- Haruno, Masahiko, and Takefumi Yamazaki. 1996. High-performance bilingual text alignment using statistical and dictionary information. In *ACL 34*, pp. 131-138.
- Hatzivassiloglou, Vasileios, and Kathleen R. McKeown. 1993. Towards the automatic identification of adjectival scales: clustering adjectives according to meaning. In *ACL 31*, pp. 172-182.
- Hawthorne, Mark. 1994. The computer in literary analysis: Using TACT with students. *Computers and the Humanities* 28:19-27.

- Hearst, Marti, and Christian Plaunt. 1993. Subtopic structuring for full-length document access. In *SIGIR '93*, pp. 59-68.
- Hearst, Marti A. 1991. Noun homograph disambiguation using local context in large text corpora. In *Seventh Annual Conference of the UW Centre for the New OED and Text Research*, pp. 1-22, Oxford.
- Hearst, Marti A. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 14*, pp. 539-545.
- Hearst, Marti A. 1994. *Context and Structure in Automated Full-Text Information Access*. PhD thesis, University of California at Berkeley.
- Hearst, Marti A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics* 23:33-64.
- Hearst, Marti A., and Hinrich Schütze. 1995. Customizing a lexicon to better suit a computational task. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 77-96. Cambridge, MA: MIT Press.
- Henderson, James, and Peter Lane. 1998. A connectionist architecture for learning to parse. In *ACL 36/COLING 17*, pp. 531-537.
- Hermjakob, Ulf, and Raymond J. Mooney. 1997. Learning parse and translation decisions from examples with rich context. In *ACL 35/EACL 8*, pp. 482-489.
- Hertz, John A., Richard G. Palmer, and Anders S. Krogh. 1991. *Introduction to the theory of neural computation*. Redwood City, CA: Addison-Wesley.
- Herwijnen, Eric van. 1994. *Practical SGML*, 2nd edition. Dordrecht: Kluwer Academic.
- Hickey, Raymond. 1993. Lexa: Corpus processing software. Technical report, The Norwegian Computing Centre for the Humanities, Bergen.
- Hindle, Donald. 1990. Noun classification from predicate argument structures. In *ACL 28*, pp. 268-275.
- Hindle, Donald. 1994. A parser for text corpora. In B.T.S. Atkins and A. Zampolli (eds.), *Computational Approaches to the Lexicon*, pp. 103-151. Oxford: Oxford University Press.
- Hindle, Donald, and Mats Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics* 19:103-120.
- Hirst, Graeme. 1987. *Semantic Interpretation and the Resolution of Ambiguity*. Cambridge: Cambridge University Press.
- Hodges, Julia, Shiyun Yie, Ray Reighart, and Lois Boggess. 1996. An automated system that assists in the generation of document indexes. *Natural Language Engineering* 2:137-160.
- Holmes, V. M., L. Stowe, and L. Cupples. 1989. Lexical expectations in parsing complement-verb sentences. *Journal of Memory and Language* 28:668-689.
- Honavar, Vasant, and Giora Slutzki (eds.). 1998. *Grammatical inference: 4th international colloquium, ICGI-98*. Berlin: Springer.
- Hopcroft, John E., and Jeffrey D. Ullman. 1979. *Introduction to automata theory, languages, and computation*. Reading, MA: Addison-Wesley.

- Hopper, Paul J., and Elizabeth Closs Traugott. 1993. *Grammaticalization*. Cambridge: Cambridge University Press.
- Hornby, A. S. 1974. *Oxford Advanced Learner's Dictionary of Current English*. Oxford: Oxford University Press. Third Edition.
- Horning, James Jay. 1969. *A study of grammatical inference*. PhD thesis, Stanford.
- Huang, T., and King Sun Fu. 1971. On stochastic context-free languages. *Information Sciences* 3:201-224.
- Huddleston, Rodney. 1984. *Introduction to the Grammar of English*. Cambridge: Cambridge University Press.
- Hull, David. 1996. Stemming algorithms - A case study for detailed evaluation. *Journal of the American Society for Information Science* 47:70-84.
- Hull, David. 1998. A practical approach to terminology alignment. In Didier Bourigault, Christian Jacquemin, and Marie-Claude L'Homme (eds.), *Proceedings of Computerm '98*, pp. 1-7, Montreal, Canada.
- Hull, David, and Doug Oard (eds.). 1997. *AAAI Symposium on Cross-Language Text and Speech Retrieval*. Stanford, CA: AAAI Press.
- Hull, David A., and Gregory Grefenstette. 1998. Querying across languages: A dictionary-based approach to multilingual information retrieval. In Karen Sparck Jones and Peter Willett (eds.), *Readings in Information Retrieval*. San Francisco: Morgan Kaufmann.
- Hull, David A., Jan O. Pedersen, and Hinrich Schütze. 1996. Method combination for document filtering. In *SIGIR '96*, pp. 279-287.
- Hutchins, S. E. 1970. *Stochastic Sources for Context-free Languages*. PhD thesis, University of California, San Diego.
- Ide, Nancy, and Jean Véronis (eds.). 1995. *The Text Encoding Initiative: Background and Context*. Dordrecht: Kluwer Academic. Reprinted from *Computers and the Humanities* 29(1-3), 1995.
- Ide, Nancy, and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics* 24:1-40.
- Ide, Nancy, and Donald Walker. 1992. Introduction: Common methodologies in humanities computing and computational linguistics. *Computers and the Humanities* 26:327-330.
- Inui, K., V. Sornlertlamvanich, H. Tanaka, and T. Tokunaga. 1997. A new formalization of probabilistic GLR parsing. In *Proceedings of the Fifth International Workshop on Parsing Technologies (IWPT-97)*, pp. 123-134, MIT.
- Isabelle, Pierre. 1987. Machine translation at the TAUM group. In Margaret King (ed.), *Machine Translation Today: The State of the Art*, pp. 247-277. Edinburgh: Edinburgh University Press.
- Jacquemin, Christian. 1994. FASTR: A unification-based front-end to automatic indexing. In *Proceedings of RIAO*, pp. 34-47, Rockefeller University, New York.
- Jacquemin, Christian, Judith L. Klavans, and Evelyne Tzoukermann. 1997. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *ACL 35/EACL 8*, pp. 24-31.

- Jain, Anil K., and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice Hall.
- Jeffreys, Harold. 1948. *Theory of Probability*. Oxford: Clarendon Press.
- Jelinek, Frederick. 1969. Fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development* pp. 675-685.
- Jelinek, Frederick. 1976. Continuous speech recognition by statistical methods. *IEEE* 64:532-556.
- Jelinek, Frederick. 1985. Markov source modeling of text generation. In J. K. Skwirzynski (ed.), *The Impact of Processing Techniques on Communications*, volume E91 of NATO ASI series, pp. 569-598. Dordrecht: M. Nijhoff.
- Jelinek, Fred. 1990. Self-organized language modeling for speech recognition. Printed in (Waibel and Lee 1990), pp. 450-506.
- Jelinek, Frederick. 1997. *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press.
- Jelinek, Frederick, Lalit R. Bahl, and Robert L. Mercer. 1975. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory* 21:250-256.
- Jelinek, F., J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi, and S. Roukos. 1994. Decision tree parsing using a hidden derivation model. In *Proceedings of the 1994 Human Language Technology Workshop*, pp. 272-277. DARPA.
- Jelinek, Fred, and John D. Lafferty. 1991. Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics* 17:315-324.
- Jelinek, F., J. D. Lafferty, and R. L. Mercer. 1990. Basic methods of probabilistic context free grammars. Technical Report RC 16374 (#72684), IBM T. J. Watson Research Center.
- Jelinek, F., J. D. Lafferty, and R. L. Mercer. 1992a. Basic methods of probabilistic context free grammars. In P. Laface and R. De Mori (eds.), *Speech Recognition and Understanding: Recent Advances, Trends, and Applications*, volume 75 of Series F: Computer and Systems Sciences. Springer Verlag.
- Jelinek, Fred, and Robert Mercer. 1985. Probability distribution estimation from sparse data. *IBM Technical Disclosure Bulletin* 28:2591-2594.
- Jelinek, Frederick, Robert L. Mercer, and Salim Roukos. 1992b. Principles of lexical language modeling for speech recognition. In Sadaoki Furui and M. Mohan Sondhi (eds.), *Advances in Speech Signal Processing*, pp. 651-699. New York: Marcel Dekker.
- Jensen, Karen, George E. Heidorn, and Stephen D. Richardson (eds.). 1993. *Natural language processing: The PLNLP approach*. Boston: Kluwer Academic Publishers.
- Johansson, Stig, G. N. Leech, and H. Goodluck. 1978. *Manual of information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital computers*. Oslo: Dept of English, University of Oslo.

- Johnson, Mark. 1998. The effect of alternative tree representations on tree bank grammars. In *Proceedings of Joint Conference on New Methods in Language Processing and Computational Natural Language Learning (NeM-LaP3/CoNLL98)*, pp. 39-48, Macquarie University.
- Johnson, W. E. 1932. Probability: deductive and inductive problems. *Mind* 41: 421-423.
- Joos, Martin. 1936. Review of *The Psycho-Biology of Language*. *Language* 12: 196-210.
- Jorgensen, Julia. 1990. The psychological reality of word senses. *Journal of Psycholinguistic Research* 19:167-190.
- Joshi, Aravind K. 1993. Tree-adjoining grammars. In R. E. Asher (ed.), *The Encyclopedia of Language and Linguistics*. Oxford: Pergamon Press.
- Justeson, John S., and Slava M. Katz. 1991. Co-occurrences of antonymous adjectives and their contexts. *Computational Linguistics* 17:1-19.
- Justeson, John S., and Slava M. Katz. 1995a. Principled disambiguation: Discriminating adjective senses with modified nouns. *Computational Linguistics* 24: 1-28.
- Justeson, John S., and Slava M. Katz. 1995b. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering* 1:9-27.
- Kahneman, Daniel, Paul Slovic, and Amos Tversky (eds.). 1982. *Judgment under uncertainty: heuristics and biases*. Cambridge: Cambridge University Press.
- Kan, Min-Yen, Judith L. Klavans, and Kathleen R. McKeown. 1998. Linear segmentation and segment significance. In *WVLC 6*, pp. 197-205.
- Kaplan, Ronald M., and Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*, pp. 173-281. Cambridge, MA: MIT Press.
- Karlsson, Fred, Atro Voutilainen, Juha Heikkilä, and Arto Anttila. 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Berlin: Mouton de Gruyter.
- Karov, Yael, and Shimon Edelman. 1998. Similarity-based word sense disambiguation. *Computational Linguistics* 24:41-59.
- Karttunen, Lauri. 1986. Radical lexicalism. Technical Report 86-68, Center for the Study of Language and Information, Stanford CA.
- Katz, Slava M. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing* ASSP-35:400-401.
- Katz, Slava M. 1996. Distribution of content words and phrases in text and language modelling. *Natural Language Engineering* 2:15-59.
- Kaufman, Leonard, and Peter J. Rousseeuw. 1990. *Finding groups in data*. New York: Wiley.

- Kaufmann, Stefan. 1998. Second-order cohesion: Using wordspace in text segmentation. Department of Linguistics, Stanford University.
- Kay, Martin, and Martin Röschkeisen. 1993. Text-translation alignment. *Computational Linguistics* 19:121-142.
- Kehler, Andrew. 1997. Probabilistic coreference in information extraction. In *EMNLP* 2, pp. 163-173.
- Kelly, Edward, and Phillip Stone. 1975. *Computer Recognition of English Word Senses*. Amsterdam: North-Holland.
- Kempe, André. 1997. Finite state transducers approximating hidden markov models. In *ACL 35/EACL 8*, pp. 460-467.
- Kennedy, Graeme. 1998. *An Introduction to Corpus Linguistics*. London: Longman.
- Kent, Roland G. 1930. Review of *Relative Frequency as a Determinant of Phonetic Change*. *Language* 6:86-88.
- Kilgarriff, Adam. 1993. Dictionary word sense distinctions: An enquiry into their nature. *Computers and the Humanities* 26:365-387.
- Kilgarriff, Adam. 1997. "i don't believe in word senses". *Computers and the Humanities* 31:91-113.
- Kilgarriff, Adam, and Tony Rose. 1998. Metrics for corpus similarity and homogeneity. Manuscript, ITRI, University of Brighton.
- Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science* 220:671-680.
- Klavans, Judith, and Min-Yen Kan. 1998. Role of verbs in document analysis. In *ACL 36/COLING 17*, pp. 680-686.
- Klavans, Judith L., and Evelyne Tzoukermann. 1995. Dictionaries and corpora: Combining corpus and machine-readable dictionary data for building bilingual lexicons. *Journal of Machine Translation* 10.
- Klein, Sheldon, and Robert F. Simmons. 1963. A computational approach to grammatical coding of English words. *Journal of the Association for Computing Machinery* 10:334-347.
- Kneser, Reinhard, and Hermann Ney. 1995. Improved backing-off for *m*-gram language modeling. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, volume 1, pp. 181-184.
- Knight, Kevin. 1997. Automating knowledge acquisition for machine translation. *AI Magazine* 18:81-96.
- Knight, Kevin, Ishwar Chander, Matthew Haines, Vasileios Hatzivassiloglou, Eduard Hovy, Masayo Iida, Steve Luk, Richard Whitney, and Kenji Yamada. 1995. Filling knowledge gaps in a broad-coverage MT system. In *Proceedings of IJCAI-95*.
- Knight, Kevin, and Jonathan Graehl. 1997. Machine transliteration. In *ACL 35/EACL 8*, pp. 128-135.

- Knight, Kevin, and Vasileios Hatzivassiloglou. 1995. Two-level, many-paths generation. In *ACL 33*, pp. 252-260.
- Knill, Kate M., and Steve Young. 1997. Hidden markov models in speech and language processing. In Steve Young and Gerrit Bloothoof (eds.), *Corpus-Based Methods in Language and Speech Processing*, pp. 27-68. Dordrecht: Kluwer Academic.
- Kohonen, Teuvo. 1997. *Self-Organizing Maps*. Berlin, Heidelberg, New York: Springer Verlag. Second Extended Edition.
- Korfhage, Robert R. 1997. *Information Storage and Retrieval*. Berlin: John Wiley.
- Krenn, Brigitte, and Christer Samuelsson. 1997. The linguist's guide to statistics. manuscript, University of Saarbrücken.
- Krovetz, Robert. 1991. Lexical acquisition and information retrieval. In Uri Zernik (ed.), *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 45-64. Hillsdale, NJ: Lawrence Erlbaum.
- Kruskal, J. B. 1964a. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29:1-27.
- Kruskal, J. B. 1964b. Nonmetric multidimensional scaling: A numerical method. *Psychometrika* 29:115-129.
- Kučera, Henry, and W. Nelson Francis. 1967. *Computational Analysis of Present-Day American English*. Providence, RI: Brown University Press.
- Kupiec, Julian. 1991. A trellis-based algorithm for estimating the parameters of a hidden stochastic context-free grammar. In *Proceedings of the Speech and Natural Language Workshop*, pp. 241-246. DARPA.
- Kupiec, Julian. 1992a. An algorithm for estimating the parameters of unrestricted hidden stochastic context-free grammars. In *COLING 14*, pp. 387-393.
- Kupiec, Julian. 1992b. Robust part-of-speech tagging using a Hidden Markov Model. *Computer Speech and Language* 6:225-242.
- Kupiec, Julian. 1993a. An algorithm for finding noun phrase correspondences in bilingual corpora. In *ACL 31*, pp. 17-22.
- Kupiec, Julian. 1993b. MURAX: A robust linguistic approach for question answering using an on-line encyclopedia. In *SIGIR '93*, pp. 181-190.
- Kupiec, Julian, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *SIGIR '95*, pp. 68-73.
- Kwok, K. L., and M. Chan. 1998. Improving two-stage ad-hoc retrieval for short queries. In *SIGIR '98*, pp. 250-256.
- Lafferty, John, Daniel Sleator, and Davy Temperley. 1992. Grammatical trigrams: A probabilistic model of link grammar. In *Proceedings of the 1992 AAAI Fall Symposium on Probabilistic Approaches to Natural Language*.
- Lakoff, George. 1987. *Women, fire, and dangerous things*. Chicago, IL: University of Chicago Press.
- Landauer, Thomas K., and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review* 104:211-240.

- Langacker, Ronald W. 1987. *Foundations of Cognitive Grammar*, volume 1. Stanford, CA: Stanford University Press.
- Langacker, Ronald W. 1991. *Foundations of Cognitive Grammar*, volume 2. Stanford, CA: Stanford University Press.
- Laplace, Pierre Simon marquis de. 1814. *Essai philosophique sur les probabilités*. Paris: Mme. Ve. Courcier.
- Laplace, Pierre Simon marquis de. 1995. *Philosophical Essay On Probabilities*. New York: Springer-Verlag.
- Lari, K., and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language* 4: 35-56.
- Lari, K., and S. J. Young. 1991. Application of stochastic context free grammars using the inside-outside algorithm. *Computer Speech and Language* 5:237-257.
- Lau, Raymond. 1994. Adaptive statistical language modelling. Master's thesis, Massachusetts Institute of Technology.
- Lau, Ray, Ronald Rosenfeld, and Salim Roukos. 1993. Adaptive language modeling using the maximum entropy principle. In *Proceedings of the Human Language Technology Workshop*, pp. 108-113. ARPA.
- Lauer, Mark. 1995a. Corpus statistics meet the noun compound: Some empirical results. In *ACL* 33, pp. 47-54.
- Lauer, Mark. 1995b. *Designing Statistical Language Learners: Experiments on Noun Compounds*. PhD thesis, Macquarie University, Sydney, Australia.
- Leacock, Claudia, Martin Chodorow, and George A. Miller. 1998. Using corpus statistics and Wordnet relations for sense identification. *Computational Linguistics* 24:147-165.
- Lesk, Michael. 1986. Automatic sense disambiguation: How to tell a pine cone from an ice cream cone. In *Proceedings of the 1986 SIGDOC Conference*, pp. 24-26, New York. Association for Computing Machinery.
- Lesk, M. E. 1969. Word-word association in document retrieval systems. *American Documentation* 20:27-38.
- Levin, Beth. 1993. *English Verb Classes and Alternations*. Chicago: The University of Chicago Press.
- Levine, John R., Tony Mason, and Doug Brown. 1992. *Lex & Yacc*, 2nd edition. Sebastopol, CA: O'Reilly & Associates.
- Levinson, S. E., L. R. Rabiner, and M. M. Sondhi. 1983. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal* 62:1035-1074.
- Lewis, David D. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR '92*, pp. 37-50.
- Lewis, David D., and Karen Sparck Jones. 1996. Natural language processing for information retrieval. *Communications of the ACM* 39:92-101.

- Lewis, David D., and Marc Ringuette. 1994. A comparison of two learning algorithms for text categorization. In *Proc. SDAIR 94*, pp. 81-93, Las Vegas, NV.
- Lewis, David D., Robert E. Schapire, James P. Callan, and Ron Papka. 1996. Training algorithms for linear text classifiers. In *SIGIR '96*, pp. 298-306.
- Li, Hang, and Naoki Abe. 1995. Generalizing case frames using a thesaurus and the mdl principle. In *Proceedings of Recent Advances in Natural Language Processing*, pp. 239-248, Tzigov Chark, Bulgaria.
- Li, Hang, and Naoki Abe. 1996. Learning dependencies between case frame slots. In *COLING 16*, pp. 10-15.
- Li, Hang, and Naoki Abe. 1998. Word clustering and disambiguation based on co-occurrence data. In *ACL 36/COLING 17*, pp. 749-755.
- Li, Wentian. 1992. Random texts exhibit Zipf's-law-like word frequency distribution. *IEEE Transactions on Information Theory* 38:1842-1845.
- Lidstone, G. J. 1920. Note on the general case of the Bayes-Laplace formula for inductive or *a priori* probabilities. *Transactions of the Faculty of Actuaries* 8: 182-192.
- Light, Marc. 1996. Morphological cues for lexical semantics. In *ACL 34*, pp. 25-31.
- Littlestone, Nick. 1995. Comparing several linear-threshold learning algorithms on tasks involving superfluous attributes. In A. Friediris (ed.), *Proceedings of the 12th International Conference on Machine Learning*, pp. 353-361, San Francisco, CA. Morgan Kaufmann.
- Littman, Michael L., Susan T. Dumais, and Thomas K. Landauer. 1998a. Automatic cross-language information retrieval using latent semantic indexing. In Gregory Grefenstette (ed.), *Cross Language Information Retrieval*. Kluwer.
- Littman, Michael L., Fan Jiang, and Greg A. Keim. 1998b. Learning a language-independent representation for terms from a partially aligned corpus. In Jude Shavlik (ed.), *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 314-322. Morgan Kaufmann.
- Losee, Robert M. (ed.). 1998. *Text Retrieval and Filtering*. Boston, MA: Kluwer Academic Publishers.
- Lovins, Julie Beth. 1968. Development of a stemming algorithm. *Translation and Computational Linguistics* 11:22-31.
- Luhn, H. P. 1960. Keyword-in-context index for technical literature (KWIC index). *American Documentation* 11:288-295.
- Lyons, John. 1968. *Introduction to Theoretical Linguistics*. Cambridge: Cambridge University Press.
- MacDonald, M. A., N. J. Pearlmutter, and M. S. Seidenberg. 1994. The lexical nature of syntactic ambiguity resolution. *Psychological Review* 101:676-703.
- MacKay, David J. C., and Linda C. Peto. 1990. Speech recognition using hidden Markov models. *The Lincoln Laboratory Journal* 3:41-62.

- Magerman, David M. 1994. *Natural language parsing as statistical pattern recognition*. PhD thesis, Stanford University.
- Magerman, David M. 1995. Statistical decision-tree models for parsing. In *ACL 33*, pp. 276-283.
- Magerman, David M., and Mitchell P. Marcus. 1991. Pearl: A probabilistic chart parser. In *EACL 4*. Also published in the Proceedings of the 2nd International Workshop for Parsing Technologies.
- Magerman, David M., and Carl Weir. 1992. Efficiency, robustness, and accuracy in Picky chart parsing. In *ACL 30*, pp. 40-47.
- Mandelbrot, Benoit. 1954. Structure formelle des textes et communication. *Word* 10:1-27.
- Mandelbrot, Benoit B. 1983. *The Fractal Geometry of Nature*. New York: W. H. Freeman.
- Mani, Inderjeet, and T. Richard MacMillan. 1995. Identifying unknown proper names in newswire text. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 41-59. Cambridge, MA: MIT Press.
- Manning, Christopher D. 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In *ACL 31*, pp. 235-242.
- Manning, Christopher D., and Bob Carpenter. 1997. Probabilistic parsing using left corner language models. In *Proceedings of the Fifth International Workshop on Parsing Technologies (IWPT-97)*, pp. 147-158, MIT.
- Marchand, Hans. 1969. *Categories and types of present-day English word-formation*. München: Beck.
- Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *ARPA Human Language Technology Workshop*, pp. 110-115.
- Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics* 19:313-330.
- Markov, Andrei A. 1913. An example of statistical investigation in the text of 'Eugene Onyegin' illustrating coupling of 'tests' in chains. In *Proceedings of the Academy of Sciences, St. Petersburg*, volume 7 of VI, pp. 153-162.
- Marr, David. 1982. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. New York: W. H. Freeman.
- Marshall, Ian. 1987. Tag selection using probabilistic methods. In Roger Garside, Geoffrey Sampson, and Geoffrey Leech (eds.), *The Computational analysis of English: a corpus-based approach*, pp. 42-65. London: Longman.
- Martin, James. 1991. Representing and acquiring metaphor-based polysemy. In Uri Zernik (ed.), *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 389-415. Hillsdale, NJ: Lawrence Erlbaum.

- Martin, W. A., K. W. Church, and R. S. Patil. 1987. Preliminary analysis of a breadth-first parson algorithm: Theoretical and experimental results. In Leonard Bolc (ed.), *Natural Language Parsing Systems*. Berlin: Springer Verlag. Also MIT LCS technical report TR-261.
- Masand, Brij, Gordon Linoff, and David Waltz. 1992. Classifying news stories using memory based reasoning. In *SIGIR '92*, pp. 59-65.
- Maxwell, III, John T. 1992. The problem with mutual information. Manuscript, Xerox Palo Alto Research Center, September 15, 1992.
- McClelland, James L., David E. Rumelhart, and the PDP Research Group (eds.). 1986. *Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models*. Cambridge, MA: The MIT Press.
- McCullagh, Peter, and John A. Nelder. 1989. *Generalized Linear Models*, 2nd edition, chapter 4, pp. 101-123. Chapman and Hall.
- McDonald, David D. 1995. Internal and external evidence in the identification and semantic categorization of proper names. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 21-39. Cambridge MA: MIT Press.
- McEnery, Tony, and Andrew Wilson. 1996. *Corpus Linguistics*. Edinburgh: Edinburgh University Press.
- McGrath, Sean. 1997. *PARSEME.1ST: SGML for Software Developers*. Upper Saddle River, NJ: Prentice Hall PTR.
- McMahon, John G., and Francis J. Smith. 1996. Improving statistical language model performance with automatically generated word hierarchies. *Computational Linguistics* 22:217-247.
- McQueen, C.M. Sperberg, and Lou Burnard (eds.). 1994. *Guidelines for Electronic Text Encoding and Interchange (TEI P3)*. Chicago, IL: ACH/ACL/ALLC (Association for Computers and the Humanities, Association for Computational Linguistics, Association for Literary and Linguistic Computing).
- McRoy, Susan W. 1992. Using multiple knowledge sources for word sense disambiguation. *Computational Linguistics* 18:1-30.
- Melamed, I. Dan. 1997a. A portable algorithm for mapping bitext correspondence. In *ACL 35/EACL 8*, pp. 305-312.
- Melamed, I. Dan. 1997b. A word-to-word model of translational equivalence. In *ACL 35/EACL 8*, pp. 490-497.
- Mel'čuk, Igor Aleksandrovich. 1988. *Dependency Syntax: theory and practice*. Albany: State University of New York.
- Mercer, Robert L. 1993. Inflectional morphology needs to be authenticated by hand. In *Working Notes of the AAAI Spring Symposium on Building Lexicons for Machine Translation*, pp. 99-99, Stanford, CA. AAAI Press.
- Merialdo, Bernard. 1994. Tagging English text with a probabilistic model. *Computational Linguistics* 20:155-171.

- Miclet, Laurent, and Colin de la Higuera (eds.). 1996. *Grammatical inference: learning syntax from sentences: Third International Colloquium, ICGI-96*. Berlin: Springer.
- Miikkulainen, Risto (ed.). 1993. *Subsymbolic Natural Language Processing*. Cambridge MA: MIT Press.
- Mikheev, Andrei. 1998. Feature lattices for maximum entropy modelling. In *ACL* 36, pp. 848-854.
- Miller, George A., and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes* 6:1-28.
- Miller, Scott, David Stallard, Robert Bobrow, and Richard Schwartz. 1996. A fully statistical approach to natural language interfaces. In *ACL* 34, pp. 55-61.
- Minsky, Marvin Lee, and Seymour Papert (eds.). 1969. *Perceptrons: an introduction to computational geometry*. Cambridge, MA: MIT Press. Partly reprinted in (Shavlik and Dietterich 1990).
- Minsky, Marvin Lee, and Seymour Papert (eds.). 1988. *Perceptrons: an introduction to computational geometry*. Cambridge, MA: MIT Press. Expanded edition.
- Mitchell, Tom M. 1980. The need for biases in learning generalizations. Technical Report Department of Computer Science. CBM-TR-117, Rutgers University. Reprinted in (Shavlik and Dietterich 1990), pp. 184-191.
- Mitchell, Tom M. (ed.). 1997. *Machine Learning*. New York: McGraw-Hill.
- Mitra, Mandar, Chris Buckley, Amit Singhal, and Claire Cardie. 1997. An analysis of statistical and syntactic phrases. In *Proceedings of RIAO*.
- Moffat, Alistair, and Justin Zobel. 1998. Exploring the similarity space. *ACM SIGIR Forum* 32.
- Mood, Alexander M., Franklin A. Graybill, and Duane C. Boes. 1974. *Introduction to the theory of statistics*. New York: McGraw-Hill. 3rd edition.
- Mooney, Raymond J. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *EMNLP* 1, pp. 82-91.
- Moore, David S., and George P. McCabe. 1989. *Introduction to the practice of statistics*. New York: Freeman.
- Morris, Jane, and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics* 17: 21-48.
- Mosteller, Frederick, and David L. Wallace. 1984. *Applied Bayesian and Classical Inference - The Case of The Federalist Papers*. Springer Series in Statistics. New York: Springer-Verlag.
- Nagao, Makoto. 1984. A framework of a mechanical translation between Japanese and English by analogy principle. In Alick Elithorn and Ranan B. Banerji (eds.), *Artificial and Human Intelligence*, pp. 173-180. Edinburgh: North-Holland.
- Neff, Mary S., Brigitte Bläser, Jean-Marc Langé, Hubert Lehmann, and Isabel Zapata Dominguez. 1993. Get it where you can: Acquiring and maintaining bilingual lexicons for machine translation. In *Working Notes of the AAAI Spring Symposium on Building Lexicons for Machine Translation*, pp. 98-98, Stanford, CA. AAAI Press.

- Nevill-Manning, Craig G., Ian H. Witten, and Gordon W. Paynter. 1997. Browsing in digital libraries: a phrase-based approach. In *Proceedings of ACM Digital Libraries*, pp. 230-236, Philadelphia, PA. Association for Computing Machinery.
- Newmeyer, Frederick J. 1988. *Linguistics: The Cambridge Survey*. Cambridge, England: Cambridge University Press.
- Ney, Hermann, and Ute Essen. 1993. Estimating 'small' probabilities by leaving-one-out. In *Eurospeech '93*, volume 3, pp. 2239-2242. ESCA.
- Ney, Hermann, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependencies in stochastic language modeling. *Computer Speech and Language* 8:1-28.
- Ney, Hermann, Sven Martin, and Frank Wessel. 1997. Statistical language modeling using leaving-one-out. In Steve Young and Gerrit Bloothoof (eds.), *Corpus-Based Methods in Language and Speech Processing*, pp. 174-207. Dordrecht: Kluwer Academic.
- Ng, Hwee Tou, and John Zelle. 1997. Corpus-based approaches to semantic interpretation in natural language processing. *AI Magazine* 18:45-64.
- Ng, Hwee Tou, and Hian Beng Lee. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *ACL 34*, pp. 40-47.
- Nie, Jian-Yun, Pierre Isabelle, Pierre Plamondon, and George Foster. 1998. Using a probabilistic translation model for cross-language information retrieval. In *WVLC 6*, pp. 18-27.
- Nießen, S., S. Vogel, H. Ney, and C. Tillmann. 1998. A DP based search algorithm for statistical machine translation. In *ACL 36/COLING 17*, pp. 960-967.
- Nunberg, Geoffrey. 1990. *The Linguistics of Punctuation*. Stanford, CA: CSLI Publications.
- Nunberg, Geoff, and Annie Zaenen. 1992. Systematic polysemy in lexicology and lexicography. In *Proceedings of Euralex II*, Tampere, Finland.
- Oaksford, M., and N. Chater. 1998. *Rational Models of Cognition*. Oxford, England: Oxford University Press.
- Oard, Douglas W., and Nicholas DeClaris. 1996. Cognitive models for text filtering. Manuscript, University of Maryland, College Park.
- Ostler, Nicholas, and B. T. S. Atkins. 1992. Predictable meaning shift: Some linguistic properties of lexical implication rules. In James Pustejovsky and Sabine Bergler (eds.), *Lexical Semantics and Knowledge Representation: Proceedings of the 1st SIGLEX Workshop*, pp. 76-87. Berlin: Springer Verlag.
- Paik, Woojin, Elizabeth D. Liddy, Edmund Yu, and Mary McKenna. 1995. Categorizing and standardizing proper nouns for efficient information retrieval. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 61-73. Cambridge MA: MIT Press.
- Palmer, David D., and Marti A. Hearst. 1994. Adaptive sentence boundary disambiguation. In *ANLP 4*, pp. 78-83.

- Palmer, David D., and Marti A. Hearst. 1997. Adaptive multilingual sentence boundary disambiguation. *Computational Linguistics* 23:241-267.
- Paul, Douglas B. 1990. Speech recognition using hidden markov models. *The Lincoln Laboratory Journal* 3:41-62.
- Pearlmutter, N., and M. MacDonald. 1992. Plausibility and syntactic ambiguity resolution. In *Proceedings of the 14th Annual Conference of the Cognitive Society*.
- Pedersen, Ted. 1996. Fishing for exactness. In *Proceedings of the South-Central SAS Users Group Conference*, Austin TX.
- Pedersen, Ted, and Rebecca Bruce. 1997. Distinguishing word senses in untagged text. In *EMNLP 2*, pp. 197-207.
- Pereira, Fernando, and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *ACL 30*, pp. 128-135.
- Pereira, Fernando, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *ACL 31*, pp. 183-190.
- Pinker, Steven. 1994. *The Language Instinct*. New York: William Morrow.
- Pollard, Carl, and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Chicago, IL: University of Chicago Press.
- Pook, Stuart L., and Jason Catlett. 1988. Making sense out of searching. In *Information Online* 88, pp. 148-157, Sydney. The Information Science Section of the Library Association of Australia.
- Porter, M. F. 1980. An algorithm for suffix stripping. *Program* 14:130-137.
- Poznański, Victor, and Antonio Sanfilippo. 1995. Detecting dependencies between semantic verb subclasses and subcategorization frames in text corpora. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 175-190. Cambridge, MA: MIT Press.
- Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. 1988. *Numerical Recipes in C*. Cambridge: Cambridge University Press.
- Procter, P. (ed.). 1978. *Longman dictionary of contemporary English*. Harlow, England: Longman Group.
- Prokosch, E. 1933. Review of selected studies of the principle of relative frequency in language. *Language* 9:89-92.
- Pustejovsky, James. 1991. The generative lexicon. *Computational Linguistics* 17: 409-441.
- Pustejovsky, James, Sabine Bergler, and Peter Anick. 1993. Lexical semantic techniques for corpus analysis. *Computational Linguistics* 19:331-358.
- Qiu, Yonggang, and H.P. Frei. 1993. Concept based query expansion. In *SIGIR '93*, pp. 160-169.
- Quinlan, J. R. 1986. Induction of decision trees. *Machine Learning* 1:81-106. Reprinted in (Shavlik and Dietterich 1990).
- Quinlan, John Ross. 1993. *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann Publishers.

- Quinlan, J. R. 1996. Bagging, boosting, and C4.5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI '96)*, pp. 725-730.
- Quirk, Randolph, Sidney Greenbaum, Geoffrey Leech, and Jan Svartvik. 1985. *A Comprehensive Grammar of the English Language*. London: Longman.
- Rabiner, Lawrence, and Bing-Hwang Juang. 1993. *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: PTR Prentice-Hall.
- Rabiner, Lawrence R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE* 77:257-286. Reprinted in (Waibel and Lee 1990), pp. 267-296.
- Ramsey, Fred L., and Daniel W. Schafer. 1997. *The statistical sleuth: a course in methods of data analysis*. Belmont, CA: Duxbury Press.
- Ramshaw, Lance A., and Mitchell P. Marcus. 1994. Exploring the statistical derivation of transformational rule sequences for part-of-speech tagging. In *The Balancing Act. Proceedings of the Workshop*, pp. 86-95, Morristown NJ. Association of Computational Linguistics.
- Rasmussen, Edie. 1992. Clustering algorithms. In William B. Frakes and Ricardo Baeza-Yates (eds.), *Information Retrieval*, pp. 419-442. Englewood Cliffs, NJ: Prentice Hall.
- Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. In *EMNLP 1*, pp. 133-142.
- Ratnaparkhi, Adwait. 1997a. A linear observed time statistical parser based on maximum entropy models. In *EMNLP 2*, pp. 1-10.
- Ratnaparkhi, Adwait. 1997b. A simple introduction to maximum entropy models for natural language processing. Technical Report IRCS Report 97-08, Institute for Research in Cognitive Science, Philadelphia, PA.
- Ratnaparkhi, Adwait. 1998. Unsupervised statistical models for prepositional phrase attachment. In *ACL 36/COLING 17*, pp. 1079-1085.
- Ratnaparkhi, Adwait, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the ARPA Workshop on Human Language Technology*, pp. 250-255, Plainsboro, NJ.
- Read, Timothy R. C., and Noel A. C. Cressie. 1988. *Goodness-of-fit statistics for discrete multivariate data*. New York: Springer Verlag.
- Resnik, Philip. 1992. Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In *COLING 14*, pp. 418-425.
- Resnik, Philip. 1996. Selectional constraints: an information-theoretic model and its computational realization. *Cognition* 61:127-159.
- Resnik, Philip, and Marti Hearst. 1993. Structural ambiguity and conceptual relations. In *WVLC 1*, pp. 58-64.
- Resnik, Philip, and David Yarowsky. 1998. A perspective on word sense disambiguation methods and their evaluation. In *Proceedings of the SIGLEX workshop Tagging Text with Lexical Semantics*, pp. 79-86, Washington, DC.
- Resnik, Philip Stuart. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships*. PhD thesis, University of Pennsylvania.

- Reynar, Jeffrey C., and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *ANLP 5*, pp. 16-19.
- Riley, Michael D. 1989. Some applications of tree-based modeling to speech and language indexing. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 339-352. Morgan Kaufmann.
- Riloff, Ellen, and Jessica Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *EMNLP 2*, pp. 117-124.
- Ristad, Eric Sven. 1995. A natural law of succession. Technical Report CS-TR-495-95, Princeton University.
- Ristad, Eric Sven. 1996. Maximum entropy modeling toolkit. Manuscript, Princeton University.
- Ristad, Eric Sven, and Robert G. Thomas. 1997. Hierarchical non-emitting Markov models. In *ACL 35/EACL 8*, pp. 381-385.
- Roark, Brian, and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *ACL 36/COLING 17*, pp. 1110-1116.
- Robertson, S.E., and K. Sparck Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27:129-146.
- Rocchio, J. J. 1971. Relevance feedback in information retrieval. In Gerard Salton (ed.), *The Smart Retrieval System - Experiments in Automatic Document Processing*, pp. 313-323. Englewood Cliffs, NJ: Prentice-Hall.
- Roche, Emmanuel, and Yves Schabes. 1995. Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics* 21:227-253.
- Roche, Emmanuel, and Yves Schabes. 1997. *Finite-State Language Processing*. Boston, MA: MIT Press.
- Roget, P. M. 1946. *Roget's International Thesaurus*. New York: Thomas Y. Crowell.
- Rosenblatt, Frank (ed.). 1962. *Principles of neurodynamics; perceptrons and the theory of brain mechanisms*. Washington, DC: Spartan Books.
- Rosenfeld, Ronald. 1994. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. PhD thesis, CMU. Technical report CMU-CS-94-138.
- Rosenfeld, Roni. 1996. A maximum entropy approach to adaptive statistical language modelling. *Computer Speech and Language* 10:187-228.
- Rosenfeld, Ronald, and Xuedong Huang. 1992. Improvements in stochastic language modeling. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 107-111. Morgan Kaufmann.
- Rosenkrantz, Stanley J., and Philip M. Lewis, II. 1970. Deterministic left corner parser. In *IEEE Conference Record of the 11th Annual Symposium on Switching and Automata*, pp. 139-152.
- Ross, Ian C., and John W. Tukey. 1975. Introduction to these volumes. In John Wilder Tukey (ed.), *Index to Statistics and Probability*, pp. iv-x. Los Altos, CA: R & D Press.

- Roth, Dan. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Menlo Park CA. AAAI Press.
- Rumelhart, D. E., and J. L. McClelland. 1986. On learning the past tenses of English verbs. In James L. McClelland, David E. Rumelhart, and the PDP Research Group (eds.), *Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models*, pp. 216-271. Cambridge, MA: The MIT Press.
- Rumelhart, David E., James L. McClelland, and the PDP research group (eds.). 1986. *Parallel Distributed Processing. Explorations in the Microstructure of Cognition. Volume 1: Foundations*. Cambridge, MA: The MIT Press.
- Rumelhart, David E., and David Zipser. 1985. Feature discovery by competitive learning. *Cognitive Science* 9:75-112.
- Russell, Stuart J., and Peter Norvig. 1995. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall.
- Sakakibara, Y., M. Brown, R. Hughey, I. S. Mian, K. Sjölander, R. C. Underwood, and D. Haussler. 1994. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Research* 22:5112-5120.
- Salton, Gerard. 1971a. Experiments in automatic thesaurus construction for information retrieval. In *Proceedings IFIP Congress*, pp. 43-49.
- Salton, Gerard (ed.). 1971b. *The Smart Retrieval System - Experiments in Automatic Document Processing*. Englewood Cliffs, NJ: Prentice-Hall.
- Salton, Gerard. 1989. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Reading, MA: Addison Wesley.
- Salton, G., J. Allan, C. Buckley, and A. Singhal. 1994. Automatic analysis, theme generation and summarization of machine-readable texts. *Science* 264:1421-1426.
- Salton, Gerard, and James Allen. 1993. Selective text utilization and text traversal. In *Proceedings of ACM Hypertext 93*, New York. Association for Computing Machinery.
- Salton, Gerard, and Chris Buckley. 1991. Global text matching for information retrieval. *Science* 253:1012-1015.
- Salton, Gerard, Edward A. Fox, and Harry Wu. 1983. Extended boolean information retrieval. *Communications of the ACM* 26:1022-1036.
- Salton, Gerard, and Michael J. McGill. 1983. *Introduction to modern information retrieval*. New York: McGraw-Hill.
- Salton, Gerard, and R. W. Thorpe. 1962. An approach to the segmentation problem in speech analysis and language translation. In *Proceedings of the 1961 International Conference on Machine Translation of Languages and Applied Language Analysis*, volume 2, pp. 703-724, London. Her Majesty's Stationery Office.
- Sampson, Geoffrey. 1989. How fully does a machine-usable dictionary cover English text? *Literary and Linguistic Computing* 4:29-35.

- Sampson, Geoffrey. 1995. *English for the Computer*. New York: Oxford University Press.
- Sampson, Geoffrey. 1997. *Educating Eve*. London: Cassell.
- Samuel, Ken, Sandra Carberry, and K. Vijay-Shanker. 1998. Dialogue act tagging with transformation-based learning. In *ACL 36/COLING 17*, pp. 1150-1156.
- Samuelsson, Christer. 1993. Morphological tagging based entirely on bayesian inference. In *9th Nordic Conference on Computational Linguistics*, Stockholm University, Stockholm, Sweden.
- Samuelsson, Christer. 1996. Handling sparse data by successive abstraction. In *COLING 16*, pp. 895-900.
- Samuelsson, Christer, and Atro Voutilainen. 1997. Comparing a linguistic and a stochastic tagger. In *ACL 35/EACL 8*, pp. 246-253.
- Sanderson, Mark, and C. J. van Rijsbergen. 1998. The impact on retrieval effectiveness of the skewed frequency distribution of a word's senses. *ACM Transactions on Information Systems*. To appear.
- Sankoff, D. 1971. Branching processes with terminal types: applications to context-free grammars. *Journal of Applied Probability* 8:233-240.
- Santorini, Beatrice. 1990. Part-of-speech tagging guidelines for the Penn treebank project. 3rd Revision, 2nd printing, Feb. 1995. University of Pennsylvania.
- Sapir, Edward. 1921. *Language: an introduction to the study of speech*. New York: Harcourt Brace.
- Sato, Satoshi. 1992. CIM: An example-based translation aid system. In *COLING 14*, pp. 1259-1263.
- Saund, Eric. 1994. Unsupervised learning of mixtures of multiple causes in binary data. In J. Cowan, G. Tesauro, and J. Alspector (eds.), *Advances in Neural Information Processing Systems 6*. San Mateo, CA: Morgan Kaufmann Publishers.
- Schabes, Yves. 1992. Stochastic lexicalized tree-adjoining grammars. In *COLING 14*, pp. 426-432.
- Schabes, Yves, Anne Abeillé, and Aravind Joshi. 1988. Parsing strategies with lexicalized grammars: Tree adjoining grammars. In *COLING 12*, pp. 578-583.
- Schabes, Yves, Michal Roth, and Randy Osborne. 1993. Parsing the Wall Street Journal with the Inside-Outside algorithm. In *EACL 6*, pp. 341-347.
- Schapire, Robert E., Yoram Singer, and Amit Singhal. 1998. Boosting and Rocchio applied to text filtering. In *SIGIR '98*, pp. 215-223.
- Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pp. 44-49, Manchester, England.
- Schütze, Carson T. 1996. *The empirical base of linguistics: grammaticality judgments and linguistic methodology*. Chicago, IL: University of Chicago Press.
- Schütze, Hinrich. 1992a. Context space. In Robert Goldman, Peter Norvig, Eugene Charniak, and Bill Gale (eds.), *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pp. 113-120, Menlo Park, CA. AAAI Press.

- Schütze, Hinrich. 1992b. Dimensions of meaning. In *Proceedings of Supercomputing '92*, pp. 787-796, Los Alamitos, CA. IEEE Computer Society Press.
- Schütze, Hinrich. 1993. Part-of-speech induction from scratch. In *ACL 31*, pp. 251-258.
- Schütze, Hinrich. 1995. Distributional part-of-speech tagging. In *EACL 7*, pp. 141-148.
- Schütze, Hinrich. 1997. *Ambiguity Resolution in Language Learning*. Stanford, CA: CSLI Publications.
- Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics* 24:97-124.
- Schütze, Hinrich, David A. Hull, and Jan O. Pedersen. 1995. A comparison of classifiers and document representations for the routing problem. In *SIGIR '95*, pp. 229-237.
- Schütze, Hinrich, and Jan O. Pedersen. 1995. Information retrieval based on word senses. In *Fourth Annual Symposium on Document Analysis and Information Retrieval*, pp. 161-175, Las Vegas, NV.
- Schütze, Hinrich, and Jan O. Pedersen. 1997. A cooccurrence-based thesaurus and two applications to information retrieval. *Information Processing & Management* 33:307-318.
- Schütze, Hinrich, and Yoram Singer. 1994. Part-of-speech tagging using a variable memory Markov model. In *ACL 32*, pp. 181-187.
- Shannon, Claude E. 1948. A mathematical theory of communication. *Bell System Technical Journal* 27:379-423, 623-656.
- Shannon, Claude E. 1951. Prediction and entropy of printed English. *Bell System Technical Journal* 30:50-64.
- Shavlik, Jude W., and Thomas G. Dietterich (eds.). 1990. *Readings in Machine Learning*. San Mateo, CA: Morgan Kaufmann.
- Shemtov, Hadar. 1993. Text alignment in a tool for translating revised documents. In *EACL 6*, pp. 449-453.
- Sheridan, Paraic, and Alan F. Smeaton. 1992. The application of morpho-syntactic language processing to effective phrase matching. *Information Processing & Management* 28:349-370.
- Sheridan, Paraic, Martin Wechsler, and Peter Schäuble. 1997. Cross language speech retrieval: Establishing a baseline performance. In *SIGIR '97*, pp. 99-108.
- Shimohata, Sayori, Toshiyuko Sugio, and Junji Nagata. 1997. Retrieving collocations by co-occurrences and word order constraints. In *ACL 35/EACL 8*, pp. 476-481.
- Siegel, Sidney, and N. John Castellan, Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences*, 2nd edition. New York: McGraw Hill.
- Silverstein, Craig, and Jan O. Pedersen. 1997. Almost-constant-time clustering of arbitrary corpus subsets. In *SIGIR '97*, pp. 60-66.

- Sima'an, Khalil. 1996. Computational complexity of probabilistic disambiguation by means of tree-grammars. In *COLING 16*, pp. 1175-1180.
- Sima'an, Khalil, Rens Bod, S. Krauwer, and Remko Scha. 1994. Efficient disambiguation by means of stochastic tree substitution grammars. In *Proceedings International Conference on New Methods in Language Processing*.
- Simard, Michel, G. F. Foster, and P. Isabelle. 1992. Using cognates to align sentences in bilingual corpora. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, pp. 67-81.
- Simard, Michel, and Pierre Plamondon. 1996. Bilingual sentence alignment: Balancing robustness and accuracy. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas (AMTA-96)*, pp. 135-144.
- Sinclair, John (ed.). 1995. *Collins COBUILD English dictionary*. London: Harper Collins. New edition, completely revised.
- Singhal, Amit, Gerard Salton, and Chris Buckley. 1996. Length normalization in degraded text collections. In *Fifth Annual Symposium on Document Analysis and Information Retrieval*, pp. 149-162, Las Vegas, NV.
- Sipser, Michael. 1996. *Introduction to the theory of computation*. Boston, MA: PWS Publishing Company.
- Siskind, Jeffrey Mark. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition* 61:39-91.
- Skut, Wojciech, and Thorsten Brants. 1998. A maximum-entropy partial parser for unrestricted text. In *WVLC 6*, pp. 143-151.
- Smadja, Frank. 1993. Retrieving collocations from text: Xtract. *Computational Linguistics* 19:143-177.
- Smadja, Frank, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics* 22:1-38.
- Smadja, Frank A., and Kathleen R. McKeown. 1990. Automatically extracting and representing collocations for language generation. In *ACL 28*, pp. 252-259.
- Smeaton, Alan F. 1992. Progress in the application of natural language processing to information retrieval tasks. *The Computer Journal* 35:268-278.
- Smith, Tony C., and John G. Cleary. 1997. Probabilistic unification grammars. In *1997 Australasian Natural Language Processing Summer Workshop*, pp. 25-32, Macquarie University.
- Snedecor, George Waddel, and William G. Cochran. 1989. *Statistical methods*. Ames: Iowa State University Press. 8th edition.
- Sparck Jones, Karen. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 28:11-21.
- Sparck Jones, Karen, and Peter Willett (eds.). 1998. *Readings in Information Retrieval*. San Francisco: Morgan Kaufmann.
- Sproat, Richard William. 1992. *Morphology and computation*. Cambridge, MA: MIT Press.

- Sproat, Richard W., Chilin Shih, William Gale, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics* 22:377-404.
- St. Laurent, Simon. 1998. *XML: A Primer*. Foster City, CA: MIS Press/IDG Books.
- Stanfill, Craig, and David Waltz. 1986. Toward memory-based reasoning. *Communications of the ACM* 29:1213-1228.
- Steier, Amy M., and Richard K. Belew. 1993. Exporting phrases: A statistical analysis of topical language. In R. Casey and B. Croft (eds.), *Second Annual Symposium on Document Analysis and Information Retrieval*, pp. 179-190, Las Vegas, NV.
- Stolcke, Andreas. 1995. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics* 21:165-202.
- Stolcke, Andreas, and Stephen M. Omohundro. 1993. Hidden Markov model induction by Bayesian model merging. In S. J. Hanson, J. D. Cowan, and C. Lee Giles (eds.), *Advances in Neural Information Processing Systems 5*, pp. 11-18, San Mateo, CA: Morgan Kaufmann.
- Stolcke, Andreas, and Stephen M. Omohundro. 1994a. Best-first model merging for hidden Markov model induction. Technical Report TR-94-003, International Computer Science Institute, University of California at Berkeley.
- Stolcke, Andreas, and Stephen M. Omohundro. 1994b. Inducing probabilistic grammars by Bayesian model merging. In *Grammatical Inference and Applications: Proceedings of the Second International Colloquium on Grammatical Inference*. Springer Verlag.
- Stolcke, A., E. Shriberg, R. Bates, N. Coccaro, D. Jurafsky, R. Martin, M. Meteer, K. Ries, P. Taylor, and C. Van Ess-Dykema. 1998. Dialog act modeling for conversational speech. In *Applying Machine Learning to Discourse Processing*, pp. 98-105, Menlo Park, CA: AAAI Press.
- Stolz, Walter S., Percy H. Tannenbaum, and Frederick V. Carstensen. 1965. A stochastic approach to the grammatical coding of English. *Communications of the ACM* 8:399-405.
- Strang, Gilbert. 1988. *Linear algebra and its applications*, 3rd edition. San Diego: Harcourt, Brace, Jovanovich.
- Strzalkowski, Tomek. 1995. Natural language information retrieval. *Information Processing & Management* 31:397-417.
- Stubbs, Michael. 1996. *Text and corpus analysis: computer-assisted studies of language and culture*. Oxford: Blackwell.
- Suppes, Patrick. 1970. Probabilistic grammars for natural languages. *Synthese* 22:95-116.
- Suppes, Patrick. 1984. *Probabilistic Metaphysics*. Oxford: Blackwell.
- Suppes, Patrick, Michael Böttner, and Lin Liang. 1996. Machine learning comprehension grammars for ten languages. *Computational Linguistics* 22:329-350.
- Tabor, Whitney. 1994. *Syntactic Innovation: A Connectionist Model*. PhD thesis, Stanford.

- Tague-Sutcliffe, Jean. 1992. The pragmatics of information retrieval experimentation, revisited. *Information Processing & Management* 28:467-490. Reprinted in (Sparck Jones and Willett 1998).
- Talmy, Leonard. 1985. Lexicalization patterns: Semantic structure in lexical form. In Timothy Shopen (ed.), *Language Typology and Syntactic Description III: Grammatical Categories and the Lexicon*, pp. 57-149. Cambridge, MA: Cambridge University Press.
- Tanenhaus, M. K., and J. C. Trueswell. 1995. Sentence comprehension. In J. Miller and P. Eimas (eds.), *Handbook of Perception and Cognition*, volume 11, pp. 217-262. San Diego: Academic Press.
- Tesnière, Lucien. 1959. *Éléments de Syntaxe Structurale*. Paris: Librairie C. Klincksieck.
- Tomita, Masaru (ed.). 1991. *Generalized LR parsing*. Boston: Kluwer Academic.
- Towell, Geoffrey, and Ellen M. Voorhees. 1998. Disambiguating highly ambiguous words. *Computational Linguistics* 24:125-146.
- Trask, Robert Lawrence. 1993. *A dictionary of grammatical terms in linguistics*. London: Routledge.
- van Halteren, Hans, Jakub Zavrel, and Walter Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *ACL 36/COLING 17*, pp. 491-497.
- van Riemsdijk, Henk, and Edwin Williams. 1986. *Introduction to the Theory of Grammar*. Cambridge, MA: MIT Press.
- van Rijsbergen, C. J. 1979. *Information Retrieval*. London: Butterworths. Second Edition.
- Velardi, Paola, and Maria Teresa Pazienza. 1989. Computer aided interpretation of lexical cooccurrences. In *ACL 27*, pp. 185-192.
- Viegas, Evelyne, Boyan Onyshkevych, Victor Raskin, and Sergei Nirenburg. 1996. From submit to submitted via submission: On lexical rules in large-scale lexicon acquisition. In *ACL 34*, pp. 32-39.
- Viterbi, A. J. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory* IT-13: 1260-269.
- Vogel, Stephan, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING 16*, pp. 836-841.
- Voutilainen, A. 1995. A syntax-based part of speech analyser. In *EACL 7*, pp. 157-164.
- Waibel, Alex, and Kai-Fu Lee (eds.). 1990. *Readings in Speech Recognition*. San Mateo, CA: Morgan Kaufmann.
- Walker, Donald E. 1987. Knowledge resource tools for accessing large text files. In Sergei Nirenburg (ed.), *Machine Translation: Theoretical and methodological issues*, pp. 247-261. Cambridge: Cambridge University Press.

- Walker, Donald E., and Robert A. Amsler. 1986. The use of machine-readable dictionaries in sublanguage analysis. In Ralph Grishman and Richard Kittredge (eds.), *Analyzing language in restricted domains: sublanguage description and processing*, pp. 69-84. Hillsdale, NJ: Lawrence Erlbaum.
- Walker, Marilyn A., Jeanne C. Fromer, and Shrikanth Narayanan. 1998. Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In *ACL 36/COLING 17*, pp. 1345-1351.
- Walker, Marilyn A., and Johanna D. Moore. 1997. Empirical studies in discourse. *Computational Linguistics* 23:1-12.
- Wang, Ye-Yi, and Alex Waibel. 1997. Decoding algorithm in statistical machine translation. In *ACL 35/EACL 8*, pp. 366-372.
- Wang, Ye-Yi, and Alex Waibel. 1998. Modeling with structures in statistical machine translation. In *ACL 36/COLING 17*, pp. 1357-1363.
- Waterman, Scott A. 1995. Distinguished usage. In Branimir Boguraev and James Pustejovsky (eds.), *Corpus Processing for Lexical Acquisition*, pp. 143-172. Cambridge, MA: MIT Press.
- Weaver, Warren. 1955. Translation. In William N. Locke and A. Donald Booth (eds.), *Machine Translation of Languages: Fourteen Essays*, pp. 15-23. New York: John Wiley & Sons.
- Webster, Mort, and Mitch Marcus. 1989. Automatic acquisition of the lexical semantics of verbs from sentence frames. In *ACL 27*, pp. 177-184.
- Weinberg, Sharon L., and Kenneth P. Goldberg. 1990. *Statistics for the behavioral sciences*. Cambridge: Cambridge University Press.
- Weischedel, Ralph, Marie Meteer, Richard Schwartz, Lance Ramshaw, and Jeff Palmucci. 1993. Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics* 19:359-382.
- Wiener, Erich, Jan Pedersen, and Andreas Weigend. 1995. A neural network approach to topic spotting. In *Proc. SDAIR 95*, pp. 317-332, Las Vegas, NV.
- Wilks, Yorick, and Mark Stevenson. 1998. Word sense disambiguation using optimized combination of knowledge sources. In *ACL 36/COLING 17*, pp. 1398-1402.
- Willett, Peter. 1988. Recent trends in hierarchic document clustering: A critical review. *Information Processing & Management* 24:577-597.
- Willett, P., and V. Winterman. 1986. A comparison of some measures for the determination of inter-molecular structural similarity. *Quantitative Structure-Activity Relationships* 5:18-25.
- Witten, Ian H., and Timothy C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory* 37:1085-1094.
- Wittgenstein, Ludwig. 1968. *Philosophical Investigations [Philosophische Untersuchungen]*, 3rd edition. Oxford: Basil Blackwell. Translated by G. E. M. Anscombe.

- Wong, S. K. M., and Y. Y. Yao. 1992. An information-theoretic measure of term specificity. *Journal of the American Society for Information Science* 43:54-61.
- Wood, Mary McGee. 1993. *Categorial Grammars*. London: Routledge.
- Woolf, Henry Bosley (ed.). 1973. *Webster's new collegiate dictionary*. Springfield, MA: G. & C. Merriam Co.
- Wu, Dekai. 1994. Aligning a parallel English-Chinese corpus statistically with lexical criteria. In *ACL 32*, pp. 80-87.
- Wu, Dekai. 1995. Grammarless extraction of phrasal examples from parallel texts. In *Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Wu, Dekai. 1996. A polynomial-time algorithm for statistical machine translation. In *ACL 34*, pp. 152-158.
- Wu, Dekai, and Hongsing Wong. 1998. Machine translation with a stochastic grammatical channel. In *ACL 36/COLING 17*, pp. 1408-1415.
- Yamamoto, Mikio, and Kenneth W. Church. 1998. Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. In *WVLC 6*, pp. 28-37.
- Yang, Yiming. 1994. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *SIGIR '94*, pp. 13-22.
- Yang, Yiming. 1995. Noise reduction in a statistical approach to text categorization. In *SIGIR '95*, pp. 256-263.
- Yang, Yiming. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval* 1:69-90.
- Yarowsky, David. 1992. Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *COLING 14*, pp. 454-460.
- Yarowsky, David. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *ACL 32*, pp. 88-95.
- Yarowsky, David. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL 33*, pp. 189-196.
- Youmans, Gilbert. 1991. A new tool for discourse analysis: The vocabulary-management profile. *Language* 67:763-789.
- Younger, D. H. 1967. Recognition and parsing of context free languages in time n^3 . *Information and Control* 10:198-208.
- Zavrel, Jakub, and Walter Daelemans. 1997. Memory-based learning: Using similarity for smoothing. In *ACL 35/EACL 8*, pp. 436-443.
- Zavrel, Jakub, Walter Daelemans, and Jorn Veenstra. 1997. Resolving PP attachment ambiguities with memory-based learning. In *Proceedings of the Workshop on Computational Natural Language Learning*, pp. 136-144, Somerset, NJ. Association for Computational Linguistics.
- Zernik, Uri. 1991a. Introduction. In *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 1-26. Hillsdale, NJ: Lawrence Erlbaum.

-
- Zernik, Uri. 1991b. Train1 vs. train2: Tagging word sense in corpus. In Uri Zernik (ed.), *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 91-112. Hillsdale, NJ: Lawrence Erlbaum.
- Zipf, George Kingsley. 1929. Relative frequency as a determinant of phonetic change. *Harvard Studies in Classical Philology* 40:1-95.
- Zipf, George Kingsley. 1935. *The Psycho-Biology of Language*. Boston, MA: Houghton Mifflin.
- Zipf, George Kingsley. 1949. *Human Behavior and the Principle of Least Effort*. Cambridge, MA: Addison-Wesley.