文本数据挖掘

（第2版）

宗成庆　夏睿　张家俊　著

清華大學出版社
北京

内 容 简 介

文本数据挖掘是通过机器学习、自然语言处理和推理等相关技术或方法，理解、分析和挖掘文本的内容，从而完成信息抽取、关系发现、热点预测、文本分类和自动摘要等具体任务的信息处理技术。本书主要介绍与文本数据挖掘有关的基本概念、理论模型和实现算法，包括数据预处理、文本表示、文本分类、文本聚类、主题模型、情感分析与观点挖掘、话题检测与跟踪、信息抽取以及文本自动摘要等，最后通过具体实例展示相关技术在实际应用中的使用方法。

本书可作为高等院校计算机、自动化、网络安全、大数据分析等专业，以及利用到文本信息处理的交叉学科（如金融财经、社会人文、生物医药等）的高年级本科生或研究生从事相关研究的入门参考书，也可供相关技术研发人员阅读和参考。

图书在版编目(CIP)数据

文本数据挖掘 / 宗成庆，夏睿，张家俊著.—2 版.—北京：清华大学出版社，2022.9
ISBN 978-7-302-61295-7

Ⅰ. ①文… Ⅱ. ①宗… ②夏… ③张… Ⅲ. ①数据采集－高等学校－教材 Ⅳ. ①TP274

中国版本图书馆 CIP 数据核字(2022)第 121954 号

责任编辑：黎 强 孙亚楠
封面设计：何凤霞
责任校对：欧 洋
责任印制：朱雨萌

出版发行：清华大学出版社
网 址：http://www.tup.com.cn, http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 **邮 编：**100084
社 总 机：010-83470000 **邮 购：**010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市君旺印务有限公司
经 销：全国新华书店
开 本：185mm×260mm **印 张：**22 **字 数：**522 千字
版 次：2019 年 5 月第 1 版 2022 年 11 月第 2 版 **印 次：**2022 年 11 月第 1 次印刷
定 价：99.00 元

产品编号：092827-01

Preface

We are living in the Big Data era. Over 80% of real-world data are unstructured, in the form of natural language text, such as books, news reports, research articles, social media messages, and webpages. Although data mining and machine learning have been popular in data analysis, most data mining methods handle only structured or semi-structured data. In comparison with mining structured data, mining unstructured text data is more challenging and will also play more essential role at turning massive data into structured knowledge. There is no wonder why we have witnessed the dramatical upsurge of research on text mining and natural language processing and their applications in recent years.

Text mining is a confluence of natural language processing, data mining, machine learning, and statistics for mining knowledge from unstructured text. There have already been multiple dedicated textbooks on data mining, machine learning, statistics, and natural language processing. However, we seriously lack textbooks on text mining that systematically introduce important topics and up-to-date methods on text mining. This book "Text Data Mining" bridges this gap nicely. It is the first textbook and also a brilliant one on text data mining, which not only introduces the foundational issues but also offers a comprehensive and state-of-the-art coverage of the important and on-going research themes on text mining. With an in-depth treatment of a wide-spectrum of text mining themes and a clear introduction to the state-of-the-art deep learning methods for text mining, it makes the book unique, timely, and authoritative. It is a great textbook for graduate students as well as a valuable handbook for practitioners working on text mining, natural language processing, data mining, machine learning and their applications.

This book is written by three pioneering researchers and highly reputed experts in the fields of natural language processing and text mining. The first author has written an authoritative and popular textbook on natural language processing, adopted as a standard textbook for university undergraduate and the first-year graduate students in China. However, this new text mining book has a completely different coverage from his NLP textbook, and offers new and complementary text mining themes. Both books can be studied independently although I would strongly encourage students working on NLP and text mining to learn both.

In this text mining book, it starts with text preprocessing, including both English and Chinese text preprocessing, and proceeds to text representation, covering vector space model and distributed representation of words, phrases, sentences and documents, both in statistical modeling and deep learning models. It then introduces feature selection methods, statistical learning methods and deep neural network methods, including multi-layer feed forward neural networks, convolutional neural networks and recurrent neural networks, for document classification. It then proceeds to text clustering, covering sample and cluster similarities, various clustering methods and clustering evaluation. After introducing the fundamental theories and methods of text mining, the book uses five chapters to cover a wide spectrum of text mining applications, including topic model which is also treated as a fundamental issue from some viewpoint but can be used independently, sentiment analysis and opinion mining, theme detection and tracking, information extraction and automated document summarization. These themes are active research frontiers in text mining, and are covered comprehensively and thoroughly, with a good balance between classical methods and recent developments, including deep learning methods.

As a data mining researcher, I have been recently deeply involved in text mining due to the need to handle the large scale of real-world data. I could not find a good text mining textbook to learn and teach no matter written in English or Chinese. It is exciting to see this book provides such a comprehensive and trendy introduction. I believe this book will benefit data science researchers, graduate students, as well as those who want to put text mining into practical applications. I love reading this book and recommend it highly to everyone who wants to learn text mining!

ACM Fellow and IEEE Fellow
Abel Bliss Professor
Department of Computer Science
University of Illinois at Urbana-Champaign

第 1 版序

我们生活在大数据时代，现实世界中 80% 以上的信息是以自然语言文本形式（如书籍、新闻报道、研究论文、社交媒体和网页等）记载的非结构化数据。尽管数据挖掘和机器学习已经成为数据分析的主要手段，但是大部分数据挖掘方法只能处理结构化的或半结构化的数据。与结构化的数据挖掘任务相比，非结构化的文本挖掘具有更大的挑战性，而且这项技术能够在将海量数据转化为结构化知识的过程中发挥巨大的作用。毫无疑问，我们已经欣喜地看到，近年来文本挖掘和自然语言处理技术研究迅速崛起，并得到了广泛应用。

文本挖掘是一门综合性的技术，涉及自然语言处理、数据挖掘、机器学习和从非结构化文本中挖掘知识的统计学方法等。目前已经有不少关于数据挖掘、机器学习和统计自然语言处理的专著和教材，但是，尚没有一部系统介绍文本挖掘重要主题和最新方法的学术专著，这本《文本数据挖掘》很好地填补了这一空缺。这是第一部，也是非常优秀的一部文本数据挖掘的教科书，它不仅介绍了文本挖掘的基础性问题，而且较为全面地阐述了当前文本挖掘研究的重要课题和最新方法。该书通过对大范围文本挖掘主题的深入分析和当前最前沿的深度学习方法的清晰介绍，使其成为一部及时、权威和特色鲜明的力作。这是一部研究生的优秀教材，也是从事文本挖掘、自然语言处理、数据挖掘、机器学习及其应用技术研究和开发的专业人员的宝贵手册。

本书由三位自然语言处理和文本挖掘领域具有较高声誉的学者完成。第一作者已经撰写和出版了一部广受欢迎的《统计自然语言处理》权威教材，被中国大陆的很多大学用作高年级本科生和一年级研究生的教科书。本书与《统计自然语言处理》的覆盖范围完全不同，它所呈现的是关于文本挖掘的新主题，是对已有著作的扩展和补充。这两本书可以分别单独学习，但我强烈地建议从事自然语言处理和文本挖掘的学生能够通读。

本书从文本预处理（包括英文的和中文的文本预处理）方法介绍开始，随后给出文本表示方法，包括向量空间模型和词汇、短语、句子及文档的分布式表示，均从统计建模和深度学习建模两个角度进行了阐述。之后针对文本分类问题介绍了特征选择方法、统计学习方法和深度神经网络方法，后者又包括多层前馈神经网络、卷积神经网络和循环神经网络。接下来是文本聚类，包括简单的类别相似性度量和各种聚类算法以及性能评价方法。在对上述文本挖掘基础理论和方法进行介绍之后，本书用 5 章介绍了文本挖掘技术的具体应用，包括主题模型（从某种角度讲它也是一种基础模型，但可以独立使用）、情感分析和观点挖掘、主题发现与跟踪、信息抽取及自动文摘。这些都是目前文本

挖掘领域活跃的前沿研究课题，本书不但给予了全面而透彻的介绍，而且在传统方法和最新进展（包括深度学习方法）之间进行了很好的平衡。

近年来由于处理大规模真实数据的需要，我作为一名数据挖掘技术的研究者，已经全身心地投入到该技术的研究中。我很难找到一本很好的既可以自学，又可以用于教学的文本数据挖掘教科书，不管是中文的还是英文的。我相信这本书将使从事数据科学研究的专家、研究生和那些有意将文本数据挖掘技术融入实际应用的人们大获裨益。我喜欢这本书，并且很愿意将其推荐给所有愿意学习文本挖掘技术的读者！

韩家炜
ACM Fellow, IEEE Fellow
伊利诺伊大学厄巴纳-香槟分校计算机科学系阿贝尔 • 布利斯特聘教授

第 2 版前言

大数据、大算力、大模型技术的快速发展极大地推动和改变着自然语言处理领域的研究和应用方式、方法，这种改变的速度远远超出了我们的预估和设想。几乎在《文本数据挖掘（第 1 版）》出版的同时，预训练语言模型逐渐兴起，并得到了快速推广和应用。随后一系列大规模预训练语言模型不断在文本数据挖掘诸多任务上取得了更强的性能，获得了广泛的成功。与此同时，我们也发现了第 1 版中的缺陷和不足，热心的读者以不同方式给我们提出了宝贵的建议。这些因素促使我们撰写了第 2 版。

第 2 版与第 1 版的主要区别体现在如下三个方面：①内容更加丰富：在第 2 版中除了增加最近几年流行的预训练语言模型（包括 BERT，GPT-3 等）以外，还增加了最后一章技术应用，通过两个应用案例将全书各章的知识点串联起来，让读者看到每一章中介绍的技术如何在实际应用中发挥作用。②对部分内容进行了整合：考虑到神经网络模型是分布式表示和深度学习方法的基础性知识，第 1 版第 3 章和第 4 章中均有涉及，部分内容略有重叠，因此，第 2 版对这两章内容进行了整合。③增加了习题：在第 2 版中各章最后增加了习题，以便于读者，尤其是学生结合每章的内容进行练习和实践。

在第 2 版的撰写过程中得到了很多同事和朋友的帮助，他们或提供素材，或与作者讨论书中的内容，或帮助作者校对书稿。他们是中科院自动化所自然语言处理团队的向露博士和赵阳博士、北京中科凡语科技有限公司技术团队、南京理工大学计算机学院博士生沈祥清等。在此谨向他们表示衷心的感谢！同时感谢在互联网上对本书第 1 版提出修改建议的热心读者们。

本书的撰写工作得到了中国科学院大学教材出版中心的资助（项目编号：YJF0812003），特此感谢！

还是那句老话，尽管作者尽了最大努力希望把这本书写好，但限于水平和时间，书中难免有诸多不足和疏漏。我们真诚地欢迎并接受读者以任何方式给予的批评指正！

作者

2021 年 12 月

第 1 版前言

随着互联网和移动通信技术的快速发展和普及应用，文本数据挖掘技术备受关注，尤其随着云计算、大数据和深度学习等一系列新技术的广泛使用，文本挖掘技术已经在众多领域（如舆情分析、医疗和金融数据分析等）发挥了重要作用，表现出广阔的应用前景。

虽然十多年前我就指导博士生开展文本分类和自动文摘等相关技术的研究，但对文本数据挖掘的整体概念并没有一个清晰的认识，只是将研究的单项技术视为自然语言处理的具体应用。韩家炜教授主笔的《数据挖掘 —— 概念与技术》和刘兵教授撰写的*Web Data Mining*等专著曾让我大获裨益，每次聆听他们的学术报告和与他们当面交谈也都受益匪浅。促使我萌生撰写这部专著念头的是中国科学院大学让我开设的“文本数据挖掘”课程。2015 年底我接受中国科学院大学计算机与控制学院的邀请，开始准备“文本数据挖掘”课程的内容设计和课件编写工作，我不得不静下心来查阅大量的文献资料，认真思考这一术语所蕴藏的丰富内涵和外延，经过几年的学习、思考和教学实践，文本数据挖掘的概念轮廓渐渐清晰起来。

夏睿和张家俊两位青年才俊的加盟让我萌生的写作计划得以实现。夏睿于 2007 年硕士毕业，以优异成绩考入中国科学院自动化研究所跟随我攻读博士学位，从事情感文本分析研究，在情感分析与观点挖掘领域以第一作者身份在国际一流学术期刊和会议上发表了一系列有影响力的论文。此外，他在文本分类与聚类、主题模型、话题检测与跟踪等多个领域都颇有见地。张家俊于 2006 年本科毕业后被免试推荐到中国科学院自动化研究所跟随我攻读博士学位，主要从事机器翻译研究，之后在多语言自动摘要、信息获取和人机对话等多个研究方向都有出色的表现。自 2016 年起他同我一道在中国科学院大学讲授“自然语言处理”课程的机器翻译、自动文摘和文本分类等部分内容，颇受学生的欢迎。仰仗两位弟子扎实的理论功底和敏锐的科研悟性，很多最新的技术方法和研究成果能够得到及时的验证和实践，并被收入本书，使我倍感欣慰。

自 2016 年初动笔，到此时收官，全书耗时两年多，当然大部分写作都是在节假日、周末和其他本该休息的时间里完成的，其间进行了无数次的修改、补充和调整，所花费的时间和精力及感受到的快乐和烦恼难以言表，正所谓“痛并快乐着”。在写作过程中和初稿完成之后，得到了很多同行专家的大力支持和帮助，他们是（以姓氏拼音顺序排列）：韩先培、洪宇、李寿山、刘康、万小军、徐康、章成志、赵鑫、周玉。他们分别审阅了部分章节的内容，提出了宝贵的修改意见和建议。另外，部分研究生和博士生也为本书的写作提供了力所能及的帮助，他们是：白赫、蔡鸿杰、丁子祥、何烩烩、金晓、李俊

杰、马聪、王乐义、向露、郑士梁、朱军楠。他们帮助作者收集整理了部分文献资料，绘制了书中的部分图表，为作者节省了宝贵的时间。在此一并向他们表示衷心的感谢!

由衷地感谢韩家炜教授对本书提出的指导性意见和建议！他能够在百忙之中为本书撰序，是我们的荣幸，不胜感激!

本书的撰写工作得到了中国科学院大学教材出版中心的资助和国家自然科学基金重点项目的资助（项目编号：61333018）。

另外，不得不说的是，由于作者的水平和能力所限，加之时间和精力的不足，书中一定存在疏漏或不足，衷心地欢迎读者给予批评指正!

宗成庆

2018 年国庆节期间

目　　录

第 1 章　绪　　论

数据挖掘（data mining）技术近年来备受关注，在快速发展的大数据时代展现了极其重要和广泛的应用前景。根据文献（Han et al., 2012）给出的广义解释，数据挖掘是指从大量数据中挖掘有趣模式和知识的过程。其中，数据源包括数据库、数据仓库、Web、其他信息存储库或动态地流入系统的数据。由于这项技术最早起源于从数据库中发现和提炼有用的知识，因此这一术语的英文通常写作 knowledge discovery in database（基于数据库的知识发现，简称“知识发现”，英文缩写为 KDD）。

互联网技术的快速发展和普及应用无时无刻不在改变着人们的思维方式和生活方式。尤其 Web 2.0 出现以后，任何人都可以随时随地以任何语言通过各种平台发布信息，互联网上急速增长的海量文本成为大数据时代无法忽视的一类重要数据。文本数据挖掘（text data mining）就是从自然语言文本中自动发现和挖掘用户感兴趣的模式、信息和知识的一种方法和技术，也常常简称为文本挖掘（text mining）。这里所说的文本包括普通 TXT 文件、doc/docx 文件、PDF 文件和 HTML 文件等各类以语言文字为主要内容的数据文件。

1.1　基 本 概 念

与广义的数据挖掘技术相比，除了解析各类文件（如 doc/docx 文件、PDF 文件和 HTML 文件等）的结构所用到的专门技术以外，文本数据挖掘的最大挑战在于对非结构化自然语言文本内容的分析和理解。这里需要强调两个方面：一是文本内容几乎都是非结构化的，而不像数据库，都是结构化的；二是文本内容是由自然语言描述的，而不是纯用数据描述的。当然，文件中含有图形、图像、表格和数据也是正常的，但文件的主体内容是文本。因此，文本数据挖掘是自然语言处理（natural language processing, NLP）、模式识别（pattern recognition）和机器学习（machine learning, ML）等相关技术密切结合的一项综合性技术。

所谓的挖掘通常带有“发现、寻找、归纳、提炼”的含义。既然需要去发现和提炼，那么，所要寻找的内容往往都不是显而易见的，而是隐蔽和藏匿在文本之中的，或者是人无法在大范围内容易发现和归纳出来的。这里所说的“隐蔽”和“藏匿”既是对计算机系统而言，也是对用户而言。但无论哪一种情况，从用户的角度看，肯定都希望系统能够直接给出所关注问题的答案和结论，而不是像传统的信息检索系统一样，针对用户输入

的关键词输出无数多可能的搜索结果，让用户自己从中分析和寻找所要的答案。粗略地讲，文本挖掘类型可以归纳成两种，一种是用户的问题非常明确、具体，只是不知道问题的答案是什么，如用户希望从大量的文本中发现某人与哪些组织机构存在什么样的关系，或者根据某人对某个事件发表的言论（文字材料）分析该人所持有的观点倾向性或情绪。另一种情况是用户只是知道大概的目的，但并没有非常具体、明确的问题，如医务人员希望从大量的病例记录中发现某些疾病发病的规律和与之相关的因素。在这种情况下，可能并非指某一种疾病，也不知道哪些因素，完全需要系统自动地从病例记录中发现、归纳和提炼出相关的信息。当然，这两种类型有时并没有明显的界限。

文本挖掘技术在国民经济、社会管理、信息服务和国家安全等各个领域中都有非常重要的应用，市场需求巨大，如对于政府管理部门来说，可以通过分析和挖掘普通民众的微博、微信、短信等网络信息，及时准确地了解民意、把握舆情；在金融或商贸领域，通过对大量的新闻报道、财务报告和网络评论等文字材料的深入分析和挖掘，预测某一时间段的经济形势和股市走向；电子产品企业可随时了解和分析用户对其产品的评价及市场反应，为进一步改进产品质量、提供个性化服务等提供数据支持；而对于国家安全和公共安全部门来说，文本数据挖掘技术则是及时发现社会不稳定因素、有效掌控时局的有力工具；在医疗卫生和公共健康领域，可以通过分析大量的化验报告、病例、记录和相关文献、资料等，发现某种现象、规律和结论等。

文本挖掘作为多项技术的交叉研究领域，起源于文本分类（text classification）、文本聚类（text clustering）和文本自动摘要（automatic text summarization）等单项技术。大约在 20 世纪 50 年代文本分类和聚类作为模式识别的应用技术崭露头角，当时主要是面向图书情报分类等需求开展研究。当然，分类和聚类都是基于文本主题和内容进行的。1958 年，H.P. Luhn 提出了自动文摘的思想（Luhn, 1958），为文本挖掘领域增添了新的内容。20 世纪 80 年代末期和 90 年代初期，随着互联网技术的快速发展和普及，新的应用需求推动这一领域不断发展和壮大。美国政府资助了一系列有关信息抽取（information extraction, IE）技术的研究项目，1987 年美国国防高级研究计划局（DARPA）为了评估这项技术的性能，发起组织了第一届消息理解会议（Message Understanding Conference, MUC）。在随后的 10 年间连续组织的 7 次评测使信息抽取技术迅速成为这一领域的研究热点。之后，情感分析与观点挖掘（sentiment analysis and opinion mining）、话题检测与跟踪（topic detection and tracking, TDT）等一系列面向社交媒体的文本处理技术相继产生，并得到快速发展。现在，这一技术领域不仅在理论方法上快速成长，在系统集成和应用形式上也不断推陈出新。

1.2 文本挖掘任务

正如前面所述，文本挖掘是一个多项技术交叉的研究领域，涉及内容比较宽泛。在实际应用中通常需要几种相关技术结合起来完成某个应用任务，而挖掘技术的执行过程通常隐藏在应用系统的背后。例如，一个问答系统（question and answering, Q&A）通常需要问句解析、知识库搜索、候选答案推断和过滤、答案生成等几个环节，而在知识库构

建的过程中离不开文本聚类、分类、命名实体识别（named entity recognition, NER）、关系抽取（relation extraction）和消歧等关键技术。因此，文本挖掘通常不是一个单项技术构成的系统，而是若干技术的集成应用。以下对几种典型的文本挖掘技术做简要的介绍。

（1）文本分类

文本分类是模式分类技术的一个具体应用，其任务是将给定的文本划分到事先规定的文本类型。例如，根据中国图书馆分类法（第 5 版）[①]，所有图书按其学科内容被划分成五大类：马列主义、毛泽东思想，哲学，社会科学，自然科学和综合性图书，并细分成 22 个基本大类。“新浪网”首页划分的内容类别包括：新闻、财经、体育、娱乐、汽车、博客、视频、房产等。如何根据一部图书或者一篇文章的内容自动将其划归为某一种类别，是一项具有挑战性的任务。

本书第 5 章详细介绍文本分类技术。

（2）文本聚类

文本聚类的目的是将给定的文本集合划分成不同的类别。通常情况下从不同的角度可以聚类出不同的结果，如根据文本内容可以将其聚类成新闻类、文化娱乐类、体育类或财经类等，而根据作者的倾向性可以将其聚成褒义类（持积极、支持态度的正面观点）和贬义类（持消极、否定态度的负面观点）等。

文本聚类和文本分类的根本区别在于：分类事先知道有多少个类别，分类的过程就是将每一个给定的文本自动划归为某个确定的类别，打上类别标签。而聚类事先不知道有多少个类别，需要根据某种标准和评价指标将给定的文档集合划分成相互之间能够区分的类别。但两者又有很多相似之处，所采用的算法和模型有较大的交集，如文本表示模型、距离函数、K-means（K-均值）算法等。

本书第 6 章详细介绍文本聚类技术。

（3）主题模型

通常情况下每一篇文章都有一个主题和几个子主题，而主题可以用一组词汇表示，这些词汇之间有较强的相关性，且其概念和语义基本一致。我们可以认为每一个词汇都通过一定的概率与某个主题相关联。反过来，也可以认为某个主题以一定的概率选择某个词汇。因此，可以给出如下简单的式子：

$$p(\text{词}_i|\text{文档}_j) = \sum p(\text{词}_i|\text{主题}_k) \times p(\text{主题}_k|\text{文档}_j)$$

由此，可以计算出文档中与某个主题相关联的词汇的概率。

为了从文本中挖掘隐藏在词汇背后的主题和概念，人们提出了一系列统计模型，称为主题模型（topic model）。

本书第 7 章详细介绍主题模型。

（4）情感分析与观点挖掘

所谓的文本情感是指文本作者所表达的主观信息，即作者的观点和态度，通常指“积极（positive）”“消极（negative）”或“中性（neutral）”三类极性。因此，情感分

① https://baike.baidu.com/item/中国图书馆图书分类法/1919634?fr=aladdin。

析（sentiment analysis）又称文本倾向性分析或观点挖掘（opinion mining），其主要任务包括情感分类（sentiment classification）和属性抽取等。情感分类可以看作文本分类的一种特殊类型，它是指根据文本所表达的观点和态度等主观信息对文本进行分类，或者判断某些（篇）文本的褒贬极性。例如，某一特殊事件发生之后（如马航 MH370 飞机失联、联合国秘书长潘基文参加中国纪念反法西斯战争胜利和抗日战争胜利 70 周年阅兵活动、朝韩领导人对话等），互联网上有大量的新闻报道和用户评论，如何从这些新闻和评论中自动了解各种不同的观点（倾向性）呢？某公司发布一款新的产品之后，商家希望从众多用户的网络评论中及时地了解用户的评价意见（倾向性）、用户年龄区间、性别比例和地域分布等，以帮助公司对下一步决策做出判断。这些都属于文本情感分析所要完成的任务。

本书第 8 章介绍情感分析与观点挖掘技术。

（5）话题检测与跟踪

话题检测（topic detection, TD）通常指从众多新闻事件报道和评论中挖掘、筛选出文本的话题，而多数人关心、关注和追踪的话题称为“热点话题”。热点话题发现（hot topic discovery）、检测和跟踪是舆情分析、社会媒体计算和个性化信息服务中一项重要的技术，其应用形式多种多样。例如，“今日热点话题”是从当日所有的新闻事件中筛选出最吸引读者眼球的报道，“2021 热门话题”则是从 2021 年全年（也可能是自 2021 年 1 月 1 日起到当时某一时刻）的所有新闻事件中挑选出最受关注的前几条新闻。

本书第 9 章介绍话题检测与跟踪技术。

（6）信息抽取

信息抽取是指从非结构化、半结构化的自然语言文本（如网页新闻、学术文献、社交媒体等）中抽取实体、实体属性、实体间的关系以及事件等事实信息，并形成结构化数据输出的一种文本数据挖掘技术（Sarawagi, 2008）。典型的信息抽取任务包括命名实体识别、实体消歧（entity disambiguation）、关系抽取和事件抽取（event extraction）。

近年来，生物医学文本挖掘（biomedical/medical text mining）技术备受关注。生物医学文本挖掘指的是专门针对生物和医学领域的文本进行的分析、发现和抽取。例如，从大量的生物医学文献中研究发现某种疾病与哪些化学物质（药物）存在关系，或从大量医生记录的病例中分析、发现某些疾病的诱因或某种疾病与其他疾病之间的关系等。与其他领域的文本挖掘相比，生物医学领域的文本挖掘面临很多特殊问题，如文本中存在大量的专用术语和医学名词，甚至还有习惯用语，包括临床上使用的一些行话或者实验室命名的一些蛋白质名称等。另外，不同来源的文本格式差异很大，如病历、化验单、研究论文、公共健康指南或手册等有很大的区别。此外，如何表示和利用生物医学领域的常识，如何获取大规模标注语料等，这些都是该领域面临的特殊问题。

另外，金融领域的文本挖掘技术也是近年来研究的一大热点。如从普通用户或监管部门的角度通过可获取的财务报告、公开报道、社交网络的用户评论等信息分析某家金融企业的运营状况和社会声誉，从企业的角度通过分析内部各类报告预警可能存在的风险，或者通过分析客户数据把控信贷风险等。

需要说明的是，信息抽取中的关系通常是指两个或多个概念之间存在的某种语义联系，关系抽取就是自动发现和挖掘概念之间的语义关系。事件抽取通常是针对特定领域

的“事件”对构成事件的元素进行抽取。这里所说的“事件”与日常人们所说的事件有所不同。日常人们所说的事件是指在什么时间、地点、发生了什么事情，所发生的事情往往是一个完整的故事，包括起因、过程和结果等很多详细的描述，而事件抽取中的“事件”往往指由某个谓词框架所表达的一个具体行为或状态。如“特朗普会见安倍晋三首相”是一个由谓词“会见”触发的事件。如果说一般人所理解的事件是一个故事的话，那么，事件抽取中的“事件”只是一个动作或状态。

本书第 10 章介绍信息抽取技术。

（7）文本自动摘要

文本自动摘要或简称自动文摘（automatic summarization）是指利用自然语言处理方法自动生成摘要的一种技术。在信息过度饱和的今天，自动文摘技术具有非常重要的用途。例如，信息服务部门需要对大量的新闻报道进行自动分类，然后形成某些（个）事件报道的摘要，推送给可能感兴趣的用户，或者某些公司、政府舆情监控部门想大致了解某些用户群体所发布言论（短信、微博、微信等）的主要内容，自动摘要技术就派上了用场。

本书第 11 章介绍文本自动摘要技术。

1.3 文本挖掘面临的困难

开展文本挖掘技术研究是一项极具挑战性的工作。一方面，自然语言处理的理论体系尚未完全建立，目前对文本的分析在很大程度上仅仅处于“处理”阶段，远未达到像人一样能够进行深度语义理解的水平。另一方面，由于自然语言是人类表达情感、抒发情怀和阐述思想最重要的工具，当人们针对某些特殊的事件或现象表述自己观点的时候，往往采用委婉、掩饰甚至隐喻、反讽等修辞手段，尤其在汉语文本中这种现象更加明显，从而使得文本挖掘面临很多特殊的困难，很多在图像识别和语音识别等其他领域能够取得较好效果的机器学习方法在自然语言处理中往往难以大显身手。归纳起来，文本挖掘的主要困难大致包括如下几点。

（1）文本噪声或非规范性表达使自然语言处理面临巨大的挑战

自然语言处理通常是文本挖掘的第一步。由于文本挖掘处理的主要数据来源是互联网，而与规范的书面语相比（如各类正式出版的新闻报刊、文学作品、政论和学术论著，以及国家和地方政府电视台、广播电台播出的正规新闻稿件等），网络文本内容存在大量的非规范表述。根据（宗成庆，2013）对互联网新闻文本进行的随机采样调查，网络新闻中词的平均长度约为 1.68 个汉字，句子平均长度为 47.3 个汉字，均短于规范的书面文本中的词长和句长。相对而言，网络文本中大量使用了口语化的甚至非规范的表述方式，尤其在网络聊天文本中非规范的表述比比皆是，如“很中国”“都是咱的福祉”“摩登萌妹子一秒变身刚刚受到表彰的车间女主管 ~”等。下面是一条典型的微博信息：

> //@XXXX: //@YYYYYY: 中国科学院大学本科招生网 bkzs.ucas.ac.cn 正式开通，本科招生简章业已公布。期待我们的母校在充实新鲜血液后能够再创辉煌！晚安，果壳大！

噪声、非规范语言现象和中英文混杂等表达形式的存在使常规自然语言处理工具的性能大幅下降，如在《人民日报》《新华日报》等规范文本上训练出来的汉语分词工具通常可以达到 95%以上的准确率，甚至高达 98%以上，但在网络文本上的性能立刻下降到 90% 以下。根据（张志琳，2014）实验的结果，采用基于最大熵（maximum entropy, ME）分类器的由字构词的汉语分词方法（character-based Chinese word segmentation），当词典规模增大到 175 万多条（包括普通词汇和网络用语）时，微博分词的性能 F_1 值只能达到 90%左右。众多汉语句法分析器（syntactic parser）在规范文本上的准确率可以达到 86%甚至更高，而在网络文本上分析准确率平均下降 13 个百分点（Petrov and McDonald, 2012）。这里所说的网络文本还不包括那些微博、微信中的对话聊天文本。

（2）歧义表达与文本语义的隐蔽性

歧义是自然语言文本中常见的现象，如英语单词“bank”既可以指银行，也可以指河岸，而汉语词汇“苹果”可以指能吃的苹果，也可以指苹果公司或其电脑、手机等品牌。另外，句法结构歧义同样大量存在，如句子“关于鲁迅的文章”既可以理解为“关于[鲁迅的文章]”，也可以理解为“[关于鲁迅] 的文章”。如何解析这种固有的自然语言歧义表达早已成为自然语言处理领域研究的基础问题，但令人遗憾的是这些问题至今没有十分奏效的处理方法，在实际网络对话文本中却又出现了大量人为的千奇百怪的“特殊表达”，例如，“木有”“坑爹”“奥特”等。

有时候说话人为了回避某些事件或人物，也会故意使用一些特殊用词或者使用英文单词代替某个词汇，如“康师傅”“国妖”“范爷”等在某个特定的时间里都有具体的所指。或者说话人故意绕弯儿，如“请问 $\times\times\times$ 的爸爸的儿子的前妻的年龄是多大?”。

请看下面的一则新闻报道：

> 张小五从警 20 多年来，历尽千辛万苦，立下无数战功，曾被誉为孤胆英雄。然而，谁也未曾想到，就是这样一位曾让毒贩闻风丧胆的铁骨英雄竟然为了区区小利铤而走险，痛恨之下昨晚在家开枪自毙。

对于任何一位正常的读者，无须多想就可以完全理解这则新闻所报导的事件，但如果基于该新闻向一个文本挖掘系统提出如下问题：张小五是什么警察？他死了没有？恐怕目前很难有系统能够给出正确的回答，因为文本中并没有直接说张小五其人是警察，而是用“从警”和“毒贩”间接地告诉读者他是一名缉毒警察，用“自毙”说明他已经自杀身亡。这种隐藏在文本中的信息需要通过深入的分析和推理技术才有可能将其挖掘出来，而这往往是困难的。

（3）样本收集和标注困难

目前主流的文本挖掘方法是基于大规模数据的机器学习方法，包括统计机器学习方法和深度学习（deep learning，DL）方法，需要大量训练样本，对于统计学习方法还需要对训练数据进行标注，而收集足够多的训练样本本身就是一件非常困难的事情。一方面，很多网络内容涉及版权或隐私权的问题而难以任意获取，更不能公开或共享；另一方面，即使能够获取一些数据，处理起来也是非常耗时费力的事情，因为这些数据往往含有大

量的噪声和乱码，格式也不统一，而且没有数据标注的标准。另外，能够收集到的数据一般属于某个特定的领域，一旦领域改变，数据收集、整理和标注工作又得重新开始，而且很多非规范语言现象（包括新的网络用语、术语等）随领域而异，且随时间而变，这就极大地限制了数据规模的扩大，从而影响了文本挖掘技术的发展。

（4）挖掘目标和结果的要求难以准确表达

文本挖掘不像其他理论问题，可以清楚地建立目标函数，然后通过优化函数和求解极值最终获得理想答案。在很多情况下，我们并不清楚文本挖掘的结果将会是什么，应该如何用数学模型清晰地描述预期想要的结果和条件。例如，我们可以从某些文本中抽取出频率较高的、可以代表这些文本主题和故事的热点词汇，但如何将其组织成以流畅的自然语言表达的故事梗概（摘要），却不是一件容易的事情。再如，我们想从某个群体大量的聊天文本中发现异常，分析其是否存在什么不良图谋，那么，如何界定“异常”？什么叫“不良图谋”？本身就是模糊的概念，很难给出明确的定义，更不可能给出精确的数学描述公式。

难以描述挖掘目标的另一个原因还在于，即使同一段文字，从不同的角度对不同的评价对象考虑可能会得出不同的结论。例如，有如下一段描述：

> “I bought an iPhone a few days ago. It was such a nice phone. The touch screen was really cool. The voice quality was clear too. Although the battery life was not long, that is ok for me. However, my mother was mad with me as I did not tell her before I bought the phone. She also thought the phone was too expensive, and wanted me to return it to the shop. ...”

从被评价的对象看，结论各异：手机整体上很好（a nice phone），尤其触摸屏很酷（really cool），通话质量也清楚（clear），但是电池寿命短（battery life was not long），价格贵（too expensive）。被评价的对象通常为商品实体（entity）及其属性（aspect），详见本书第 8 章的介绍。而从不同的评价角度看，结论也不一样：站在“我（I）”的角度，这是一部很好的手机，“我”很喜欢；但站在“母亲”的角度，她非常不喜欢这部手机，因为太贵了（my mother was mad with me）。

由此可见，文本挖掘与具体分析的对象和所处的角度密切相关，不能一概而论。

（5）语义表示和计算模型不甚奏效

如何有效地构建语义计算模型是长期困扰自然语言处理和计算语言学（computational linguistics）领域的一个基础问题。自深度学习方法兴起以来，词向量（word vector）表示和基于词向量的各类计算方法在自然语言处理中发挥了重要作用。但是，自然语言中的语义毕竟与图像中的像素不一样，像素可以精确地用坐标和灰度描述，而如何定义和表征词汇的语义，如何实现从词汇语义到短语语义和句子语义，最终构成段落语义和篇章语义的组合计算，始终是语言学家、计算语言学家和从事人工智能研究的学者们共同关注的核心问题之一。迄今为止，还没有一种具有较好解释性、被广泛接受且有效的语义计算模型和方法。目前大多数语义计算方法，包括众多词义消歧方法、基于主题模型的词义归纳方法和词向量组合方法等，都是基于统计的概率计算

方法，从某种意义上讲统计方法就是选择大概率事件的“赌博方法”，无论在什么情况下，只要概率大，就会成为最终被选择的答案。这实际上是一种凭经验猜谜式的权宜之计，由于计算概率的模型是基于训练样本建立起来的，而实际情况（测试集）未必都与训练样本的情况完全一致，这就必然使部分小概率事件成为“漏网之鱼”，因此，一律用概率来衡量的“赌博方法”只能解决大部分容易被统计出来的问题，却无法解决那些不易被发现、出现频率低的小概率事件，而那些小概率事件往往都是难以解决的困难问题，也就是文本挖掘面临的最大“敌人”。

综上所述，文本挖掘汇集了自然语言处理、机器学习和模式分类等各个领域的难题于一身，甚至有时候需要与图形、图像和视频理解以及真伪辨识等技术相结合，是一项综合性的应用技术。这一领域的理论体系尚未建立，而应用前景极其广阔，且时不我待，因此文本挖掘必将成为一个备受瞩目的研发热地，并将伴随相关技术的发展而迅速成长壮大。

1.4 方法概述与本书的内容组织

正如 1.1 节所述，文本挖掘属于自然语言处理、模式分类和机器学习等相关技术的交叉研究领域，因此其技术方法的使用和发展轨迹也随着相关技术的发展和变迁而改变。

回顾半个多世纪的发展历史，概括地讲，文本挖掘方法大致可以分为知识工程方法和统计学习方法两种类型。在 20 世纪 80 年代之前，文本挖掘以知识工程方法为主，这与当时基于规则的自然语言处理方法、句法模式识别和以逻辑推理方法为主导的专家系统占据主流地位的历史轨迹相吻合。这类方法的基本思路是由领域专家基于给定文本集合的经验知识和常识，人工提取和设计逻辑规则，通过推理算法对文本进行分析和挖掘。这种方法的优点是可以利用专家的经验和常识，推理的每一步都有明确的依据，最终结果有很好的解释性，但是问题在于需要耗费大量的人类资源分析和总结经验知识，系统的性能受到专家知识库（规则、词典等）的约束，一旦需要将系统移植到新的领域和任务上时，很多经验知识无法重用，系统移植周期长。到了 90 年代以后，随着统计机器学习方法的快速发展和广泛应用，基于统计机器学习的文本挖掘方法在准确率和稳定性等方面具有明显的优势，而且不需要长期占用人工资源，尤其在网络大数据时代，面对海量文本，人工手段无论在速度，还是处理数据的规模和覆盖面等各个方面显然无法与机器相比，因此统计机器学习方法逐渐成为这一领域的主流。近年来兴起的深度学习方法，或称基于神经网络的机器学习（neural network based ML）方法属于同一类方法，这类方法也可统称为数据驱动方法（data driven methods）。统计学习方法也有自身的缺陷，如有指导的（supervised），或称有监督的机器学习方法需要大量的人工标注样本，而无指导的（unsupervised）模型性能通常都比较差，而且无论是有指导的还是无指导的统计学习方法，系统最终产生的结果都缺乏充分的可解释性。

总体而言，知识工程方法和数据驱动的方法各有利弊，因此在实际应用中系统开发人员往往将两者结合起来，在某些环节利用知识工程方法，而在某些技术模块中使用统计学习方法，通过两种方法的融合尽量使系统达到较高的性能。从技术的成熟度看，知

识工程方法相对成熟，其性能的天花板也是可以预见的，而统计学习方法随着已有模型不断改进，新的模型不断提出，模型和算法的性能逐渐得到改善，而且仍有很大的上升空间，尤其在大规模数据处理方面拥有不可替代的优势，因此统计学习方法方兴未艾。这也是本书将内容重心放在统计学习方法上的原因所在。

本书主要介绍文本挖掘的基本方法和模型思路，而不涉及具体系统的实现细节，也不对具体应用领域的任务需求和面临的特殊问题给予过多的阐述，如近年来生物医药领域和金融领域的文本挖掘技术备受关注，面向这些领域需要很多领域相关的技术和资源，如领域知识库、领域相关数据的标注工具和标注样本等。作者希望本书介绍的基本方法和模型具有一定的通用性和普适性，读者掌握这些基本理论方法之后，能够根据自己面对的具体任务需求进行方法扩展和系统实现。

除了本章内容之外，后面 11 章的内容按如下思路组织，见图 1.1。

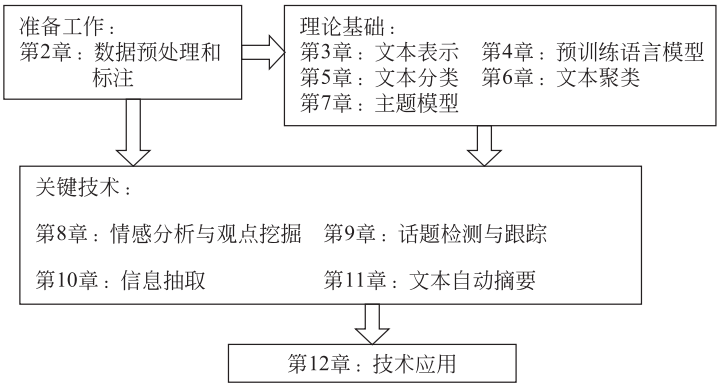


图 1.1 本书的内容组织

第 2 章介绍数据预处理和标注方法。数据预处理是后续所有模型和算法实现之前的准备阶段，如汉语、日语、越南语等文本的词语切分，尤其对于网络文本来说，文本中含有大量的噪声和非规范表达，如果不对这些数据进行预处理，后续的模型和算法必将受到干扰，很难达到预期的效果，甚至无法运行。第 3 章文本表示（text representation）和第 4 章预训练语言模型是后续几章的基础，如果不能准确地表示文本，就无法运用后面各章介绍的数学模型。第 5 章介绍的文本分类方法、第 6 章介绍的文本聚类算法和第 7 章介绍的主题模型从某种意义上讲是其他文本挖掘技术的理论基础，因为分类和聚类是模式识别最基础、最核心的两个问题，也是统计机器学习和统计自然语言处理中最常用的两种方法，后续几章介绍的模型和方法大都可以被归结为分类和聚类问题，或者采用分类或聚类的思想解决。所以，第 4 章 ∼ 第 7 章可以看作全书内容的理论基础，或称基础模型。需要说明的是，文本分类、聚类和主题模型除了作为基础模型以外，有时也被作为一种具体应用。

第 8 章 ∼ 第 11 章可以看作文本挖掘关键技术。某一项技术可以针对某个特定任务构建一个系统，也可以是几项技术联合完成一系列任务。在实际应用中，多数情况下不是单个技术的应用，而是多项相关技术的联合应用和集成。例如，在医药领域的文本挖掘任务通常涉及文本自动分类和聚类、主题模型、信息抽取和自动文摘等技术，而在面

向社交网站的舆情监控任务中，可能涉及文本分类、聚类、主题模型、话题检测与跟踪，以及情感分析和观点挖掘等，甚至还涉及自动文摘。第 12 章给出两个应用示范。作者希望通过这两个应用实例简要介绍文本挖掘技术如何在实际应用中解决问题，以满足不同用户的需求。

随着互联网和移动通信技术的快速发展和普及，很可能会出现新的应用需求和归属于文本数据挖掘的新技术，但是，我们认为不管什么样的应用需求，也无论被冠以什么名称的新技术，可能会有新的文本表示方法和类别距离计量方法，也可能会有新的实现方法和模型（如端到端（end-to-end）的神经网络模型），但聚类和分类的基本思想及其在各种任务里的渗透和应用，不会发生根本性的改变。正所谓“万变不离其宗”。

1.5 进一步阅读

本书后续各章分别介绍不同任务的文本挖掘方法，以任务目标为导向阐述各种文本挖掘任务的目标、解决思路和实现方法。本章作为全书的开篇，主要介绍文本挖掘的基本概念和面临的问题。关于数据挖掘概念的详细阐述，读者可以参阅如下文献：（Han et al., 2012）、（程显毅 等，2010）、（李雄飞 等，2010）和（毛国君 等，2007）等。（吴信东 等，2013）介绍了数据挖掘领域的十大经典算法。（Aggarwal, 2018）是一部比较全面的介绍文本数据挖掘技术的专著，通过对比读者可以发现，在该书中将文本数据挖掘看作机器学习技术的具体应用，侧重于从机器学习方法（尤其是传统的机器学习方法）的角度探讨文本信息处理问题，深度学习和神经网络方法涉及较少，并且对文本挖掘各项任务的相关工作介绍也都以传统方法为主，而近年来出现的基于深度学习方法的相关工作介绍的不多，如在文本分类、情感分析与观点挖掘中几乎都没有提及。而在本书中，我们将文本数据挖掘看作自然语言处理技术的实际应用，因为文本是自然语言的一种呈现方式，既然要从文本中挖掘用户所需要的信息，当然离不开自然语言处理技术。因此，本书以任务需求为驱动，从自然语言处理的视角通过实例和过程化描述阐述文本数据挖掘模型和算法的基本原理，如在文本表示一章中，分别从词、句子、文档的粒度归纳了基于深度学习方法的文本表示和建模方法，并且在后面的文本挖掘各项任务中，除了介绍传统的经典方法以外，都特别关注了近年来备受推崇的深度学习方法。

如果说（宗成庆，2013）是自然语言处理技术入门的一本基础性专著或教材的话，那么，本书则是一本自然语言处理技术应用的导论性读物。前者主要介绍自然语言处理的基本概念、基础理论、工具和方法，而本书重点阐述自然语言处理应用系统的实现方法和经典模型。

有些专著对某些文本数据挖掘的专项技术进行了详细阐述，具有很好的参考价值，如（Liu, 2011；2012；2015）对网络数据挖掘、情感分析和观点挖掘等概念和相关技术给予了详细介绍；（Marcu, 2000）和（Inderjeet, 2001）对自动摘要技术有详细的阐述，尤其是对早期文摘技术的介绍。在后续各章的“进一步阅读”中都会给出相关的推介。

另外需要说明的是，本书默认读者已经具备一定程度的模式识别和机器学习基础，因此对很多基础理论和方法并不做详细的介绍，略去了很多模型和公式的详细推导，只

是将其作为工具引用。如果读者想了解关于模式分类和机器学习等基础模型和公式的详细推导过程，推荐读者参阅如下专著：（李航，2019）、（周志华，2016）、（于剑，2017）、（张学工，2016）等。

习　　题

1.1　请通过例子对比知识挖掘（KDD）与文本数据挖掘之间的差异。

1.2　分析互联网上各种文本的差异，根据其结构化或规范化的程度进行分类比较。

1.3　请给出 2~3 个文本数据挖掘技术实际应用的例子。

1.4　收集一组微信或手机短信文本，分析这些文本的特点，总结一下你从这些文本中挖掘某些信息时利用了哪些特征？采用了什么方法？这些方法能否被形式化？

1.5　如果有一批外文的语料，可以先将其翻译成中文，然后再进行挖掘分析，也可以在原文的基础上先进行挖掘分析，完成之后再将结果翻译成中文。请讨论分析两种不同做法的利弊。

第 2 章　数据预处理和标注

2.1　概　　述

正如第 1 章所述，在实际应用中有监督的统计学习方法是目前构建实用系统的主流方法，而大规模带标注的数据是这种方法实现的基础和前提。在网络大数据时代，海量文本、图像和视频等各类数据都可以轻易获得。但是，直接从网上获取的数据或者来自其他渠道的原始数据，如医生书写的病历，各种微信、微博等聊天记录，各种会议记录、政府文件等，很多聊天记录和经语音识别后的文本往往都含有噪声，存在大量的非规范语言现象，这就为后续任务的模型学习造成了很大的障碍，因此在执行后续具体的挖掘任务之前通常需要对输入数据进行预处理。另外，有些后续的任务模型也需要利用文本中的某些特征信息，如词性或短语的类型、n 元词组、词之间的依存关系等。那么，通过什么工具获取这些信息呢？或者说，某些信息和问题是采用什么方法处理的呢？这就需要一些基本的自然语言处理工具。

本章简要介绍网络数据爬取、预处理和标注以及汉语自动分词、词性标注、句法分析和语言模型等基础工具和方法。

2.2　数 据 获 取

针对不同的数据挖掘任务，数据获取渠道和方式有所不同。从数据来源的渠道考虑，通常有两种情况：一种是开放域的，如面向社交媒体构建舆情检测系统时，数据自然来自所有能够获取的公共社交网络，包括移动终端，尽管文本的主题可能是关于某个或某些特定的话题，但是数据来源却是公开的；另一种是封闭域的，如面向金融领域的文本数据挖掘任务处理的数据是来自银行等金融行业的专有数据，而面向医院的数据挖掘任务处理的文本存在于医院的医疗机构内部的专用网络，普通用户是无法获取的。当然，所谓的开放域和封闭域都不是绝对的，或者在实际系统实现时，仅仅依靠某个领域内的数据是不够的，因为领域内的数据主要包含的是专业领域知识和数据，而很多常识往往存在于公共文本中，因此需要从公网上（包括维基百科、百度百科等）或教科书、专业文献中获取和补充。相对而言，来自专用网络平台的数据比较规范，而公共网络平台（尤其是社交网站）上的数据含有较多的噪声和非规范语言现象，因此需要花费更多时间进行数据的清理和预处理。

下面以获取电影评论为例，说明数据获取的一般方法。

在获取数据之前首先得知道所需要的数据一般存在于哪些网站上。“豆瓣电影”[①]提供用户对电影的评论，主页内有很多电影的链接，如图 2.1 所示（2018 年）。以《碟中谍 6》为例，这部电影的主页内部有很多评论，如图 2.2 所示，一共有 22.08 余万人给予了点评，平均分数为 8.2 分（见图 2.2 右边的“豆瓣评分”）。这个主页的下面也会提供一些评论内容及其得分，如图 2.3 所示，但不够全面，点击图中最上面一行的“全部 79481 条”之后可以查看全部短评的链接[②]。每个页面最后都会有一个“后页 >”的按钮，点击后得到下一页评论内容。通过使用 Python 的 urllib2 库可以下载一个链接所包含的数据。

图 2.1 豆瓣电影主页

图 2.2 《碟中碟 6》主页面[③]

① https://movie.douban.com/。

② https://movie.douban.com/subject/26336252/comments?status=P。

③ https://movie.douban.com/subject/26336252/?from=showing。


碟中谍6：全面瓦解的短评 · · · · · · (全部 79481 条)

我要写短评

热门 / 最新 / 好友

甄鬯鬯 看过 ★★★★☆ 2018-07-13 3047 有用

第四部的噱头是迪拜塔，第五部是徒手扒飞机，这一部则是HALO跳伞和雪山里的直升飞机。有阿汤哥在，动作场景一定拼到没毛病，大超加盟算是对了，打斗戏份排得很妙，拳拳到肉质感很好。这一部女性角色非常抢眼，一众女主女配气场惊人各有千秋，Julia的出现倒是还真对得起她这个角色，并没敷衍。

Erik Li 看过 ★★★★★ 2018-07-26 923 有用

这一集，几位女性真是太棒了，包括巴黎的那位小女警。班治和路德的老梗玩得很溜，笑到牙疼。有这样的team，阿汤哥就算七十岁都nothing impossible。阿汤哥的跑跳爬，还有无死角的驾驶技术，是越来越牛逼了。这第6集，不是全面瓦解，是全面巩固。

次等水货 看过 ★★★★☆ 2018-08-25 930 有用

那个法国女警察在一分钟之内就会爱上阿汤哥吧，太撩了。

素昔 看过 ★★★★★ 2018-07-26 752 有用

四点五分没问题，比上周的摩天大楼好看也就五倍吧。

Departure陆离 看过 ★★★★☆ 2018-07-26 456 有用

阿汤哥演的伊森太完美了，以至于白寡妇这个角色被衬托的特别好，她第一眼望过去眼神里就充满了想上伊森的情欲ㄖ。不得不说IMF的任务一次比一次变态，阿汤哥快60的身体不知道下一步还扛不扛得住（另外按照往常惯例这部里他居然不是长发真的惊了），朱莉亚回归太煽情。

图 2.3 《碟中谍 6》评论页面 ①

值得注意的是，通常这个短评网站只能访问 10 页，超过 10 页的数据需要用户登录才可以访问，一般有两种解决方案：①每部电影只抓取前 10 页内容，更多地抓取不同电影的影评数据；②使用爬虫对网站进行模拟登录，主要思路是分析人工登录网页时的信息流走向，通过爬虫模拟人工登录的过程。有的网站针对豆瓣网给出了模拟登录的方法和 Python 实战纪要，或者总结了 Python 模拟登录的一般方法，可供读者参考。

使用 Python 编程语言对某个网站进行数据抓取时，首先要查看并遵守该网站的 Robot 协议，该协议定义了网站的哪些数据可以被抓取，哪些不能抓取。图 2.4 给出了豆瓣的 Robot 协议内容，协议中的“Disallow”限定了不能被抓取的内容（很多搜索相关的内容都不能被抓取），同时规定了抓取时两次访问的时间间隔为 5 秒（即“#Crawl-delay:5”）。其中并没有约束不能抓取的影评内容，因此这部分内容是可以获取的，但是要符合抓取时间间隔的规定。其次，在抓取过程中，应尽量降低抓取的频次，实际上每次抓取都是对网站服务器的一次访问，如果抓取过于频繁，必然会影响网站服务器的正常运行。另外，应尽量在网站访问流量较少时进行抓取（如夜间），以免干扰网站的正常工作。

下载之后的网页数据一般都有较好的结构，可以通过 Python 的 Beautiful Soup 工具包对下载的网页进行解析，提取网页中的内容，并获取下一页的链接。解析网页时需要将网页的行分隔符（“\r”,“\n”）删除，网页数据中可能有很多类似于“ ”“<”的特定符号，分别表示空格和小于号等，不需要时可以将其替换掉。常见的网页特殊符号对应情况见表 2.1。

① https://movie.douban.com/subject/26336252/?from=showing。

```
User-agent: *
Disallow: /subject_search
Disallow: /amazon_search
Disallow: /search
Disallow: /group/search
Disallow: /event/search
Disallow: /celebrities/search
Disallow: /location/drama/search
Disallow: /forum/
Disallow: /new_subject
Disallow: /service/iframe
Disallow: /j/
Disallow: /link2/
Disallow: /recommend/
Disallow: /trailer/
Disallow: /doubanapp/card
Sitemap: https://www.douban.com/sitemap_index.xml
Sitemap: https://www.douban.com/sitemap_updated_index.xml
# Crawl-delay: 5

User-agent: Wandoujia Spider
Disallow: /
```

图 2.4 豆瓣的 Robot 协议 ①

表 2.1 网页中常见的特殊符号对应表

显示结果	描述	实体名称	实体编号
	空格		
<	小于号	&lt;	&#60;
>	大于号	&gt;	&#62;
&	和号	&amp;	&#38;
"	引号	&quot;	&#34;
'	撇号	' (IE 不支持)	&#39;
¢	分 (cent)	&cent;	&#162;
£	镑 (pound)	&pound;	&#163;
¥	元 (yen)	&yen;	&#165;
€	欧元 (euro)	&euro;	&#8364;
§	小节	&sect;	&#167;
©	版权 (copyright)	&copy;	&#169;
®	注册商标	&reg;	&#174;
™	商标	&trade;	&#8482;
×	乘号	&times;	&#215;
÷	除号	&divide;	&#247;

得到评论内容后还需要进行数据清理，删除噪声或者是过短的评论（通常没有意义），具体过程如下：

（1）噪声处理：抓取到的中文文本中可能会有一些英文的评论，或者在抓取英文数据时有一些其他语言的文本。这就需要对字符串的语言类型进行识别，可以借助 Python 的 langdetect 工具包帮助识别，删除那些不需要的语言数据。另外，抓取到的微博数据

① https://www.douban.com/robots.txt。

中可能含有广告链接和“@”等，需要做特殊处理。链接类可以直接删除，“@”后面一般会跟用户名，可以利用规则或模板等简单的方法判断后删除。

（2）繁体字转换：抓取到的中文文本中可能会有一些繁体字，需要将其转换成简体字，可以借助开源工具包 OpenCC[①]或其他工具完成。

（3）删除过短的评论：对于英文的评论，可以直接利用空格统计评论文本的词汇数，对于中文文本，需要使用分词工具对评论进行分词之后统计词汇数目，当然也可以简单地统计字的数目。通常删除词或字数量少于某个阈值（如 5）的评论。

（4）标签对应：不同网站上提供的标签类别不尽相同，而类别数目与希望使用的分类器也会有所差异，因此需要进行标签或类别对应。例如，从网站上抓取的评价打分是 5 分制，而情感分类器可能只需区分褒、贬两类，因此需要把不同打分的评论标签对应到“褒义”或“贬义”两个类别上，如将得分为 4 分和 5 分的样本作为褒义样本，得分为 1 分和 2 分的样本作为贬义样本，而删除那些得分为 3 分的“中立”样本。如果要学习一个褒义、中性和贬义的三类分类器，那么就可以将那些得分为 3 分的样本标注为中性。

对于其他任务的开放领域数据获取方法大同小异，只是后续的标注方法各不相同，如文本自动摘要、信息抽取等，需要人工标注的内容远比简单地标记类别复杂得多。

2.3 数据预处理

数据获取之后，通常还需要对文本进一步做预处理，主要任务包括：

（1）词条化（tokenization）：是指将给定的文本切分成为词汇单位的过程。西方语言（如英语等）天然使用空格作为词的分隔符，因此只需利用空格或标点就能实现词条化，而汉语、日语、朝鲜语（韩语）和越南语等书写中没有词语分隔标记，因此需要先进行词语切分，这一过程在中文信息处理中称作汉语自动分词（Chinese word segmentation, CWS）。

（2）去停用词：停用词（stop words）主要指功能词（functional words），通常指在各类文档中频繁出现的、附带极少实际含义的助词、介词、连词、语气词等高频词和系动词，如英文中的 the，is，at，which，on 等，汉语中的“的”“了”“是”等，这些词出现频率很高，但对于文本区分没有实质性意义，因此为了减少模型规模，节省存储空间，提高运行效率，通常在文本表示时就将这些停用词过滤掉。在具体实现时通常建立一个停用词表，在特征抽取时直接删除停用词表中的词。

（3）词形规范化：在针对西方语言的文本挖掘任务中，需要对一个词的不同形态进行归并，即词形规范化，从而提高文本处理的效率，同时减缓离散特征表示可能造成的数据稀疏问题。词形规范化过程包含两个概念，一是词形还原（lemmatization），即把发生形态变化之后的词汇还原成为原形（能够表达完整的语义），如将 cats 还原为 cat，did 还原为 do 等；二是词干提取（stemming），即去除词缀得到词根的过程，去除后的词干不一定能够表达完整的语义，如将 fisher 转换为 fish，effective 转换为 effect。

① https://opencc.byvoid.com/。

词形规范化过程一般通过规则或正则表达式实现。波特词干提取算法（Porter stemming algorithm）是一种广泛使用的英语词干提取算法[①]，采用基于规则的实现方法（Porter, 1980）。该算法主要包括如下 4 步：①将字母分为元音和辅音；②利用规则处理以 -s，-ing 和 -ed 为后缀的单词；③设计专门的规则处理复杂的后缀（如 -ational 等）；④利用规则微调处理结果。下面给出该算法的基本流程。

输入：一个英文单词；
输出：输入单词的词干或原形。
算法描述：

第 1 步：利用如下规则区分元音字母和辅音字母：
（1）字母 a, e, i, o, u 为元音；
（2）字母 y 有如下 3 种情况：
①如果 y 是单词的开头，判断为辅音，如在单词 young 中，y 是辅音；
②如果 y 的前一个字母为元音，y 被判断为辅音，如在单词 boy 中，y 是辅音字母；
③如果 y 的前一个字母为辅音，y 被判断为元音，如在单词 fly 中，y 为元音字母。
（3）除了 a, e, i, o, u, y 的其他字母均为辅音字母。

第 2 步：利用如下规则处理以 -s、-ing 和 -ed 为后缀的单词：
（1）以 -s 结尾的单词分如下几种情况处理：
①如果单词以 -sses 结尾，将其还原为 -ss，如单词 caresses 应还原为 caress；
②如果单词以 -ies 结尾，删除 -es，如 cries 变为 cri；
③如果单词以 -s 结尾，并且 s 之前的所有字母至少有一个为元音字母，考虑如下两种情况：
（a）如果该元音字母在结尾的 s 之前，则单词不变，如单词 gas 就是原形，无须变动；
（b）否则，删除尾端的字母 s，如 gaps 还原为 gap。
（2）以 -ing 结尾的单词，并且单词除了 -ing 之外，前面部分包含一个元音字母，那么，删除 -ing，如单词 doing 还原为 do。

第 3 步：利用如下规则处理其他后缀的单词：
（1）如果单词以 -y 结尾，并且 -y 前面的部分包含了元音字母，那么，将 -y 改为 i，如单词 happy 被改写为 happi；
（2）如果单词以 -ational 结尾，并且 -ational 前面的部分包含元音字母，那么，将 -ational 改写为 ate，如单词 relational 被改写为 relate。

第 4 步：利用规则微调：
对于以 -e 结尾的单词，如果该单词除去首字母和尾字母之后，其他部分包含的辅音字母个数大于 1，则去掉尾端字母 e，如 relate 被改为 relat。

算法 2.1 Porter 词干提取算法

在上述 Porter 词干提取算法中，第 2 步至第 4 步中只是给出了部分主要的改写规则，其余情况没有一一陈列，只是以此为例说明算法的基本原理。该算法的详细描述可见如下网页：

http://snowball.tartarus.org/algorithms/english/stemmer.html

① https://tartarus.org/martin/PorterStemmer/。

算法的在线测试网址为：

http://facweb.cs.depaul.edu/mobasher/classes/csc575/porter.html

算法的实现代码可从以下网页获取：

https://tartarus.org/martin/PorterStemmer/

另外，Python 的 NLTK 工具包也提供了该算法的调用函数。

需要说明的是，词干提取结果并没有统一的标准，对于同一种语言的词汇不同的词干提取算法可能给出不同的结果。除了 Porter 算法以外，Lovins stemmer（Lovins, 1968）和 Paice stemmer（Paice, 1990）也是常用的英语词干提取算法。对于其他语言，通常会参照英文的处理方式，结合语言自身的特点，专门建立针对特定语言的词干提取算法。

2.4 数 据 标 注

数据标注是有监督的机器学习方法赖以实现的基础。一般而言，数据标注的规模越大、质量越高、覆盖范围越广，处理模型的性能越好。对于不同的数据挖掘任务，数据标注的标准和规范不同，复杂程度也不一样。例如，对于文本分类任务而言，只需要对每个文档标记类别标签，而对于某些复杂任务，需要标记的信息要多得多。例如，针对电子病例分析任务，需要标注出病例中每一个“实体”的边界、类型以及与其他“实体”之间的关系。这里所说的“实体”既包括通常我们所说的命名实体（人名、地名、组织机构名、时间、数字等），也包括很多医疗领域的专用名词，如疾病、有某种症状、无某种症状、发生的频率、恶化因素、无关因素、程度等。请看如下两个例子：

①患者于【1971 年】$_{\text{Time}}$ 因时有【尿痛】$_{\text{Sym}}$ 在【当地医院】$_{\text{Hosp}}$ 检查，自述【尿液检查】$_{\text{Test}}$ 发现【尿红细胞阳性】$_{\text{TR}}$，【白细胞阳性】$_{\text{TR}}$，其余化验检查结果不详，诊断“【肾炎】$_{\text{Dis}}$”，予【“链霉素”静滴】$_{\text{Treat}}$ 治疗，后长期间断【口服中药】$_{\text{Treat}}$ 治疗。

②既往【高血压】$_{\text{Dis}}$ 病史【30 年】$_{\text{Dur}}$，【冠心病】$_{\text{Dis}}$ 病史【8 年】$_{\text{Dur}}$，【2009 年 9 月】$_{\text{Time}}$ 行【冠状动脉造影】$_{\text{TR}}$ 检查，于【前降支放置 1 枚支架】$_{\text{Treat}}$，目前偶有【胸闷】$_{\text{Sym}}$ 发作。

其中，标签 Time 表示时间，Sym 表示有这种症状，Hosp 表示医院名称，Test 表示化验检查，TR 表示化验检查结果，Dis 表示疾病名称，Treat 表示治疗方法，Dur 表示持续时间。

在电子病例分析任务中，通常会定义 20 多种不同的标签。具体标注时，一般需要开发一个标注工具，除了标注出所有“实体”的边界和类型以外，还要标注出它们之间的关系。对于上述例①，我们的标注工具给出的是图 2.5 所示的关系图。

当然，这种关系图只是为了方便标注者和领域专家直观地检查和标注，实际上系统内部存储的是特定的符号标记。例如，如果系统采用 BIO 标记法，“B”表示实体的开始，“I”表示当前“字”属于该实体，“O”表示当前“字”不属于该实体（这里的“字”可

以是任何语言单位，包括汉字、符号、标点和数字等），那么，上面例子中的第一个子句对应如下标注序列：

患/O 者/O 于/O 1/B-Time 9/I-Time 7/I-Time 1/I-Time/ 年/I-Time/因/O 时/O 有/O 尿/B-Sym 痛/I-Sym 在/O 当/B-Hosp 地/I-Hosp 医/I-Hosp 院/I-Hosp 检/O 查/O

一个实体起始于 B 标记的“字”，紧随其后的被标记为 I 的任何“字”都属于始于 B 的同类实体，而终止于非 I 标记“字”。对于这种需要专业知识指导的标注任务，如果没有领域专家的指导是很难完成的。

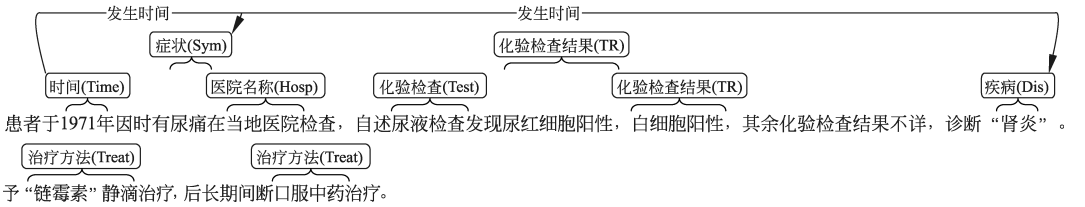


图 2.5 病例标注示例

针对多模态自动摘要方法研究，我们标注了一批包含文本、图像、音频和视频信息在内的多模态自动摘要数据。不同于同步的多模态数据（如电影），该数据集由异步多模态数据构成，即图片与文本中的句子或者视频与语句之间均不构成一一对应关系。该数据集以中英文新闻主题为中心，围绕同一个主题有多个新闻文档、新闻配图，对于每个主题都给出了限定字数的中英文文本摘要。

在数据收集时，我们选取了当时近 5 年的中英文新闻主题各 25 个，如埃博拉病毒、抗议“萨德”反导系统、李娜澳网夺冠等。对于每个主题，我们收集了同一个时间段的 20 篇新闻文档和 5~10 段视频，并确保收集到的新闻文本长度没有悬殊差异，文本一般不超过 1000 个汉字（英文词），视频不超过 2 分钟。其主要原因是，如果文本过长或视频过长，会严重增加人工标注的难度，有可能导致不同人给出的结果差异性太大。

数据标注时，我们参考了文档理解会议（Document Understanding Conference, DUC）和文本分析会议（Text Analysis Conference, TAC）的标注原则。我们聘请了 10 名研究生进行语料标注，要求他们首先阅读同一个主题的新闻文档和视频新闻，然后独立撰写摘要。撰写摘要的原则为：①确保摘要保留了新闻文档和视频新闻的重要信息；②避免摘要中出现冗余信息；③具有良好的可读性；④满足字数限制（中文摘要不超过 500 个汉字，英文摘要不超过 300 个英文词）。

每个主题最终保留三个由不同标注人独立撰写的摘要，作为参考答案。

目前大多数自动摘要系统输出的文摘形式都是文本，考虑到图文并茂的形式能够更好地提升用户体验，我们也标注了由文本和图片两种模态形式输出的摘要数据。标注这批数据时涉及文本摘要的撰写和图片的选取两项任务。关于文本摘要的撰写要求与前面介绍的方法并没有什么不同。为了完成图片选取，每个主题我们邀请两名研究生各自独立地标注出最重要的三幅图片，然后让第三位标注者综合前两位标注者给出的结果选

出三幅图片，作为最终的标准答案。选取图片的基本原则是：①与新闻的主题密切相关；②与文本摘要的内容密切相关。

上述自动文摘语料已经发布在如下网站上：http://www.nlpr.ia.ac.cn/cip/dataset.htm，有兴趣的读者可以下载使用。

综上所述，数据标注是一件费时、费力的事情，往往需要投入大量的人力和财力，因此数据共享尤为重要。本节介绍的方法和例子只是众多文本数据挖掘任务中的基本做法，在具体系统实现时需要更多详细的标注规范、标准和说明，对于很多复杂的标注任务，开发方便好用的标注工具是标注大规模数据的基本保障。

2.5 基 本 工 具

正如前面所述，文本挖掘涉及自然语言处理、模式分类和机器学习等多种技术，属于具有明确应用目标的多技术交叉研究领域。无论是前面介绍的数据预处理和数据样本标注，还是实现后面介绍的某些数据挖掘方法，通常都需要用到很多基础性的技术和方法，如在文本表示时需要对汉语文本进行词语切分、对句子进行句法分析（syntactic parsing）、词性标注（part-of-speech tagging）和语块分析（chunking）等。下面对部分技术方法和工具做简要介绍。

2.5.1 汉语自动分词与词性标注

汉语自动分词的主要任务是将汉语文本自动切分成词序列。由于词是自然语言中具有独立含义的最小的语言单位，而汉语文本中词与词之间有分隔标记，因此，词语切分是汉语文本处理的第一步。关于汉语自动分词方法，国内外有大量的研究工作，从早期的基于词典的分词方法（如最大匹配方法、最短路径分词方法等），到基于 n 元语法（n-gram）的统计切分方法，再到后来的由字构词的汉语分词方法（character-based Chinese word segmentation）等，人们先后提出了数十种切分方法。其中，由字构词的分词方法是汉语分词研究中一种标志性的创新方法，其基本思路是：句子中的任何一个单位，包括字、标点、数字和字母等（统称为“字”）在词中的位置只有 4 种可能：词首字（记为 B）、词尾字（记为 E）、词中字（记为 M）和单字词（记为 S）。B，E，M 和 S 称为词位标记。B 和 E 总是成对出现。情况见如下例子：

原始句子：特朗普在白宫会见安倍晋三。

分词结果：特朗普/ 在/ 白宫/ 会见/ 安倍晋三/ 。

用词位标记表示的分词结果：特/B 朗/M 普/E 在/S 白/B 宫/E 会/B 见/E 安/B 倍/M 晋/M 三/E 。/S

这样汉语分词问题转化为序列标注（sequence labeling）问题，可以借助大规模训练样本训练分类器完成分词任务。在实际应用中，人们也尝试将这些方法融合或集成起来，如基于 n-gram 的生成式方法与由字构词的区分式方法相结合（Wang et al., 2012），由字构词的切分方法与神经网络方法相结合等，以建立性能更好的分词系统。

在基于统计方法（包括神经网络）的自然语言处理中，对于处理单元的选择也有一定的技巧。最初研究人员以单词作为处理单元，为了降低计算的复杂度，仅会保留训练语料中高于一定频次的单词，剩余的低频词会以生词（unknown word, UNK）代替。尽管上述做法能够降低计算复杂度，但是生词导致的数据稀疏问题和语义不连贯问题是影响系统性能的重要因素，对于训练语料规模不是很大的低资源语言，问题尤其严重。而如果以字母为单位，语义的不确定性和过长的序列又是一个显而易见的问题。因此，在实际系统实现时通常采用这种方案：以子词为处理单元。所谓的子词是指介于词与字母之间的语言单位。对于屈折语，可以直接采用基于字节对编码（byte pair encoding, BPE）算法获得子词序列。该算法是 1994 年由 Philip Gage 提出来的（Gage, 1994），最初用于文本压缩，所以也被称为双字节编码压缩算法，其基本思路是统计单词范围内两两邻近的字节对，将字节对出现次数最多的进行合并，作为一个单元进行下一轮统计，重复这一过程，直到没有可合并的单元为止。以如下英语句子（语料）为例：

that fat cat is on the mat

以字母为统计单位：t h a t/ f a t/ c a t/i s/o n/ t h e/ m a t（斜杠 “/” 只是为了表明邻近字节的统计是在单词范围内统计的）。

统计两两邻近字节在整个语料中出现的次数。a 和 t 字节对出现了 4 次，出现次数最多，将其合并后：

th at/ f at/ c at/i s/o n/ t h e/ m at

之后，在前面合并后的基础上重新统计，t 和 h 对出现了 2 次，出现次数最多，因此将其合并：

th at/ f at/ c at/i s/o n/th e/ m at

到此为止，没有可进一步合并的邻近字节对，终止压缩，th, at, f, c, i, s, o, n, e, m 为最终的子词词汇表。当然，在实际处理中，由于子词压缩是在较大规模的语料上进行的，最终的子词词汇表不会像这个例子一样有很大比例的单个字母。

对于汉语文本，系统同样会面临 UNK 的问题。为了缓解 UNK 对应用任务造成的负面影响，通常会在自动分词结果的基础上再进行子词压缩。当然，也是在词的边界范围内进行压缩。具体实现方法在此不多赘述，有兴趣的读者可参阅即将出版的（宗成庆等，2022）。

词性标注是指自动为句子中的每个词打上词性类别标签，如句子“天空是蔚蓝的。”被分词和加注词性后为“天空/NN 是/NV 蔚蓝/AA 的/Aux。/PU”。符号 NN 是名词标记，VV 是动词标记，AA 是形容词标记，Aux 是结构助词标记，PU 是标点符号的标记。词性标注是句法分析的基础，词性信息是文本表示的重要特征，对于命名实体识别、关系抽取和文本情感分析等都具有重要的帮助。

词性标注是一个典型的序列标注问题，对于汉语文本来说，词性标注与自动分词有着密切的联系，因此，在很多汉语自动分词工具中都将这两项任务集成在一起，甚至采用一个模型一体化完成，如基于隐马尔可夫模型（hidden Markov model, HMM）的自动分词方法。

2.5.2 句法分析

句法分析包括短语结构分析（constituent parsing 或 phrase structure parsing）和依存关系分析（dependency parsing）。短语结构分析的目的是自动分析出句子的短语结构关系，输出句子的句法结构树（syntactic structure tree）。依存关系分析的目的则是自动分析出句子中词汇之间的语义依存关系。例如，图 2.6 是句子“警方已到现场，正在详细调查事故原因”的短语结构树，图 2.6 中的节点标记 VV，NN，ADVP，NP，VP，PU 分别是词性符号和短语标记。IP 是句子的根节点标记。图 2.7 是该句子对应的依存关系树。

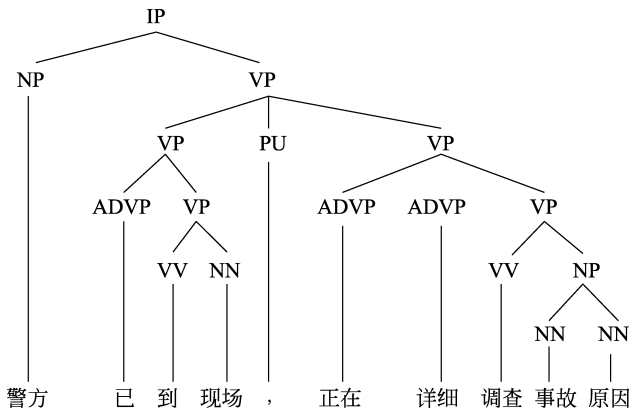


图 2.6 短语结构树示例

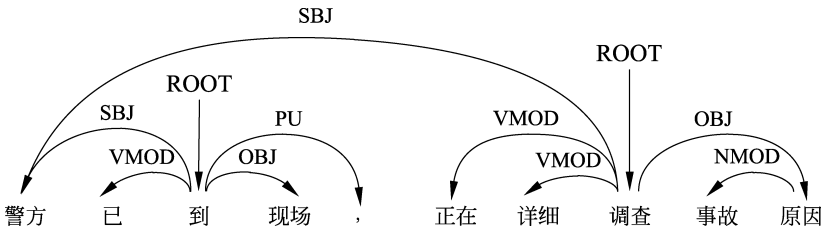


图 2.7 依存关系树示例

图 2.7 中的箭头表示依存（或支配）关系，箭头起始端为支配词，箭头指向端为被支配词。有向弧上的标记表示依存关系的类型，SBJ 表示主语关系，即箭头指向端的词是箭头起始端的词的主语。OBJ 表示宾语关系，即箭头指向端的词是箭头起始端的词的宾语。VMOD 表示动词修饰关系，即箭头指向端的词修饰箭头起始端的动词。NMOD 是名词修饰关系，即箭头指向端的词修饰箭头起始端的名词。ROOT 表示子句的根节点，PU 表示子句的标点符号。

一个句子的短语结构树可以被一一对应地转换为依存关系树，转换的基本思路是：首先确定句子的核心谓词，作为句子的唯一根节点，然后定义中心词抽取规则，抽取每个短语的中心词，非中心词受中心词的支配。

在自然语言处理中，通常将短语结构分析工具称为句法分析器，将依存关系分析工具称为依存分析器（dependency parser）。

如果句法分析器是针对一个完整的句子进行句法分析，最终希望获得句子完整的

分析树，则其分析过程被称为完全句法分析（full parsing）。在实际应用中，有时并不需要获得一个句子的完整句法分析结果，而只需要识别出句子中所包括的基本名词短语（base NP）或者基本动词短语（base VP），例如，句子“外资企业在中国经济中也发挥了重要作用”中包含基本名词短语“外资企业”“中国经济”“重要作用”和基本动词短语“发挥”。识别句子中特定类型短语的分析技术称为浅层句法分析（shallow parsing）。目前使用较多的浅层句法分析方法类似于由字构词的分词方法，标记单位可以是词，也可以是字。词位标记可以采用 B，E，M，S 四种标记法，也可以采用 B，I，O 三类标记法，如 NP-B 表示基本名词短语的首词（字），NP-I 表示该词（字）属于该名词短语，NP-O 表示该词（字）不属于该名词短语。

2.5.3 n 元语法模型

n 元语法（n-gram）（有时也称 n 元文法）是传统的语言模型（language model，LM），在自然语言处理中发挥了非常重要的作用，其基本思想是：对于一个由 l（l 为自然数，$l \geqslant 2$）个基元构成的字符串（短语、句子或片段）$s = w_1 w_2 \cdots w_l$，其概率可以用如下公式计算：

$$p(s) = p(w_1)p(w_2 \,|w_1\,)p(w_3 \,|w_1 w_2\,)\cdots p(w_l \,|w_1 w_2 \cdots w_{l-1}) = \prod_{i=1}^{l} p(w_i \,|w_1 w_2 \cdots w_{i-1}\,) \tag{2.1}$$

这里所说的“基元”可以是字、词、标点、数字或构成句子的其他任何符号，或者是短语、词性标记等，为了表述方便统称为“词”。在公式（2.1）中，意味着产生第 i（$1 \leqslant i \leqslant l$）个词的概率是由前面（按文字的书写顺序“前面”通常指左边）已经产生的 $i-1$ 个词 $w_1 w_2 \cdots w_{i-1}$ 决定的。随着句子长度的增加，条件概率的历史数目呈指数级增长。为了简化计算的复杂性并提高可行性，假设当前词的概率只与前 $n-1$（n 为整数，$1 \leqslant n \leqslant l$）个词有关。于是，式 (2.1) 变为

$$p(s) = \prod_{i=1}^{l} p(w_i \,|w_1 w_2 \cdots w_{i-1}) \approx \prod_{i=1}^{l} p(w_i \,|w_{i-n+1}, \cdots, w_{i-1}\,) \tag{2.2}$$

当 $n = 1$ 时，出现在第 i 位上的词 w_i 的概率独立于前面已经出现的词，句子是由独立的词构成的序列，这种计算模型通常称为一元文法模型，记作 unigram，或 uni-gram，或 monogram，每个词都是一个一元文法。当 $n = 2$ 时，出现在第 i 位上的词 w_i 的概率只与它前面的一个词 w_{i-1} 有关，这种计算模型称为二元文法模型。两个邻近的同现词称作二元文法，记作 bigram 或 bi-gram。例如，对于句子“We helped her yesterday”，如下词序列：We helped、helped her、her yesterday 都是二元文法。在这种情况下，句子可以看作由二元文法构成的序列链，称作一阶的马尔可夫链（Markov chain）。依此类推，当 $n = 3$ 时，出现在第 i 位置上的词 w_i 的概率只与它前面的两个词 $w_{i-2}w_{i-1}$ 有关（$i \geqslant 2$），这种计算模型称为三元文法模型。三个邻近的同现词构成的序列称作三元文法，记作 trigram 或 tri-gram。由三元文法构成的序列可以看作 2 阶的马尔可夫链，等等。

在计算 n 元语法模型时，面临的一个重要问题是如何进行数据平滑（date smoothing），以避免零概率事件（n 元语法）带来的问题。为此，人们先后提出了加 1 法（additive

smoothing）、减值法或称折扣法（discounting），以及删除插值法（deleted interpolation）等若干数据平滑方法。同时，为了消除来自不同领域、不同主题和不同类型的训练样本对模型性能产生的影响，人们也提出了若干语言模型自适应方法，在此不再一一陈述，有兴趣的读者可参阅（Chen and Goodman, 1998）和（宗成庆，2013）等。

关于汉语分词、词性标注、句法分析方法和 n 元语法模型，在很多自然语言处理专著中都有详细的介绍，这里不再多述。

由于传统的 n 元语法模型是基于离散符号（基元）进行概率计算的，无法通过相似语义的基元替换近似处理未见“基元”（“生词”）带来的数据稀疏问题，而且在后续任务中基于离散符号的统计模型远不及基于分布式向量表示的神经网络模型，因此，近年来神经网络语言模型（neural network language model, NNLM）备受推崇。NNLM 通常被简称为神经语言模型（neural language model, NLM）。本书第 3 章将对神经语言模型进行简要介绍。

2.6 进一步阅读

除了上面提到的部分自然语言处理技术之外，词义消歧（word sense disambiguition, WSD）、语义角色标注（semantic role labeling, SRL）和文本蕴涵（textual entailment）等都有可能对文本数据挖掘有所帮助，只是目前的性能尚未达到较高的水平（例如，对于规范文本的语义角色标注准确率只有 70%～80%）。相关技术方法在很多自然语言处理论著中都有描述，有兴趣的读者可以参阅（Manning and Schütze, 1999）、（Jurafsky and Martin, 2000）、（宗成庆，2013）等自然语言处理专著，这里不再赘述。

习 题[①]

2.1 使用网络爬虫程序从相关网站上获取尽量多的语料，实现程序去除语料中的噪声。

2.2 尝试使用公开的汉语分词工具，在不同文本上测试分词工具的切分性能。

2.3 分别从社交媒体平台上（如微博、QQ、Twitter 等）和正规媒体网站上（如《人民日报》等）收集尽量多的语料，详细对比不同来源语料表达形式的差异。例如，统计分析不同语料中的词汇量、平均词长、句子长度等，以及句型、命名实体、指代使用等差异。对于汉语语料，可以对比不同语料词语切分的准确率等。

2.4 实现子词压缩算法，分别对英语和汉语文本进行子词压缩实验。对于汉语文本，对比分析对分词结果进行子词压缩前后的差别。

2.5 收集部分中文的医学专业文本，使用分词工具进行词语切分，分析并校对切分结果，标注出文本中的命名实体和类型。

2.6 了解句法分析器的功能，尝试使用某开源的句法分析器分析部分句子的短语结构和词汇依存关系，思考分析结果对下游文本挖掘任务可能的用途。

① 在完成本章习题时，应遵守数据获取规范和隐私保护法，尊重别人的知识产权。

第 3 章　文 本 表 示

3.1　概　　述

文本是由文字和标点组成的字符串。字或字符组成词、词组或短语，进而形成句子、段落和篇章。要使计算机能够高效处理真实文本，就必须找到一种理想的形式化表示方法。这种表示一方面要能够真实地反映文本的内容，包括文本的主题、领域、结构和语义等，另一方面又要对不同文本有较好的区分能力，而且便于计算或处理。

文本的本质是由字符构成的字符串。字符串是无结构化的数据，但是字符串具有语法，通过语法组织起来的字符串背后隐藏着丰富的含义，这些含义无法被机器学习模型直接使用，因此首先需要将真实的文本转化为机器学习算法易于处理的表示形式。机器学习方法首先将输入的文本进行形式化，将其表示为向量或者其他形式，并基于形式化表示进行机器学习模型的训练和决策。这种将文本进行形式化的过程称为文本表示（text representation）。

文本表示方法基本上是伴随着自然语言处理和文本数据挖掘的范式迁移得到不断发展的。在文本数据挖掘技术刚刚兴起的阶段，规则方法是主流。例如，针对文本的关键词挖掘，通常采用一些字符串直接匹配的方法，这一阶段的文本表示方法以独立的字符串表示为主。基于统计机器学习方法的文本数据挖掘技术兴起以后，以向量空间模型为核心的文本表示方法成为主流，无论是词语、句子还是文档，都将其表示为词表规模的向量，从而方便了文本之间的计算。例如，在文本聚类任务中，以向量空间模型表示每个文本，利用向量之间的距离计算方法度量文本之间的相似度，从而完成相似文本的聚类。近年来，深度学习技术逐渐主导了文本数据挖掘领域，文本表示方法也从基于离散符号的高维向量空间模型过渡到基于低维连续实数向量空间的分布式表示。而且，与基于离散符号统计的向量空间模型不同，分布式文本表示往往需要与深度学习模型联合学习，才能获得高质量的文本表示。

本章首先介绍向量空间模型，然后重点介绍词语、短语、句子和文档等不同粒度语言单位的分布式表示方法。

3.2　向量空间模型

3.2.1　向量空间模型的基本概念

向量空间模型（vector space model，VSM）是一种最简单的文本表示方法。该方法

由 G. Salton 等人于 20 世纪 60 年代末期在信息检索领域中提出 (Salton et al., 1975)，最早用于 SMART 信息检索系统中，逐渐成为文本挖掘中最常用的一种表示模型。

在具体介绍 VSM 之前，我们首先给出几个相关的基本概念。

- 文本（text）：指具有长度不少于一个词的文档片段，如短语、句子、段落或整个篇章。

- 特征项（feature term）：是 VSM 中最小的不可再分的语言单元，可以是字、词、词组、短语等。在 VSM 中，一段文本被看成是由特征项组成的集合，表示为 $(t_1, t_2, \cdots, t_n)$，其中 t_i 表示第 i 个特征项。

- 特征权重（feature weight）：对于含有 n 个特征项的文本，每个特征项 t 都依据一定的原则被赋予一个权重 w，表示它们在文本中的重要性和相关性。这样，一个文本就可以用特征项及其对应的权重的集合表示：$(t_1 : w_1, t_2 : w_2, \cdots, t_n : w_n)$，简记为 $(w_1, w_2, \cdots, w_n)$。

向量空间模型假设文档符合以下两条约定：①各 t_i 互异（即没有重复）；②各 t_i 无先后顺序关系。我们可以把 $t_1, t_2, \cdots, t_n$ 看成是一个 n 维正交坐标系，那么，一个文本就可以表示为 n 维空间中的一个向量，其坐标值为 $(w_1, w_2, \cdots, w_n)$。通常我们将 $\boldsymbol{d} = (w_1, w_2, \cdots, w_n)$ 称为文本 $\boldsymbol{d}$ 在向量空间模型下的表示。如图 3.1 所示，文本 $\boldsymbol{d}_1$ 和 $\boldsymbol{d}_2$ 分别表示为向量空间中的两个 n 维向量。

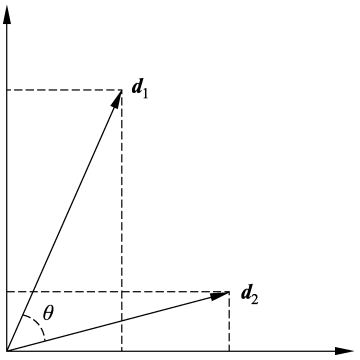


图 3.1 向量空间模型

构建向量空间模型的过程需要解决两个问题：一是如何构造特征项；二是如何计算特征项的权重。

3.2.2 特征项的构造与权重

在基于向量空间模型建立文本表示之前，通常需要依据本书第 2 章所述的词条化、去停用词、词形规范化等预处理技术，对给定文档进行规范和约减，将文档转化为词项的序列，然后定义文本表示的特征项，特征项构造好之后，向量空间就确定了，最后通过特征权重（feature weight）计算方法将每个文档表示为向量空间的一个向量表示。

首先，向量空间模型需要一个特征项集合 $(t_1, t_2, \cdots, t_n)$。如果使用词作为特征项，特征项的集合可以看作一个词表（vocabulary），此时特征项也称为词项。这个词表可以

从语料集中产生，也可以从外部导入，我们将其形象地称为词袋（bag of words，BOW），向量空间模型被称作词袋模型。

其次，如何定义特征项的权重 $(w_1, w_2, \cdots, w_n)$。该权重为向量的每个维度赋予一个值。常见的特征项权重包括下列几种：

- 布尔（BOOL）权重：表示该特征项是否在当前文本中出现，如果出现，则记为 1，否则记为 0。特征项 t_i 在文本 $\boldsymbol{d}$ 中的布尔权重记为

$$\text{bool}_i = \begin{cases} 1, & \text{如果 } t_i \text{ 在文本 } \boldsymbol{d} \text{ 中} \\ 0, & \text{否则} \end{cases} \tag{3.1}$$

- 特征频率（term frequency, TF）：表示该特征项在当前文本中出现的次数。TF 权重假设高频特征包含的信息量高于低频特征的信息量，因此在文本中出现次数越多的特征项，其重要性越大。通常用下式表示：

$$\text{tf}_i = N(t_i, \boldsymbol{d}) \tag{3.2}$$

少数高频词如采用绝对词频权重会远高于平均权重，这样并不利于文本表示，为了降低这种影响，还可以采用对数词频权重进行文本表示：

$$f_i = \log(\text{tf}_i + 1) \tag{3.3}$$

为了表示简单，如果不作特殊说明，本书中的对数计算均以 2 为底。

- 倒文档频率（inverse document frequency, IDF）权重：文档频率（document frequency，DF) 表示语料中包含特征项的文档的数目。一个特征项的 DF 越高，其包含的有效信息量往往越低。IDF 是反映特征项在整个语料中重要性的全局性统计特征，定义如下：

$$\text{idf}_i = \log \frac{N}{\text{df}_i} \tag{3.4}$$

其中，df_i 表示特征项 t_i 的 DF 值，N 是语料中的文档总数。

- 特征频率-倒文档频率（TF-IDF）权重：定义为 TF 和 IDF 的乘积：

$$\text{tf\_idf}_i = \text{tf}_i \cdot \text{idf}_i \tag{3.5}$$

TF-IDF 认为，对区别文本最有意义的特征项应该是那些在当前文本中出现频率足够高，而在文本集合的其他文本中出现频率足够低的词语。

在图 3.2 中，我们以词表作为特征项并采用 TF 权重建立向量空间模型，对左侧文档（“人工 智能 是 计算机 科学 的 一个 分支，它 企图 生产 出 一种 能 以 人类 智能 相似 的 方式 做出 反应 的 智能 机器”）进行文本表示。词表包括如下词汇：教育、智能、人类、体育、足球、运动会、AI、科学、文本、人工、计算机等，每个词汇左边对应的数字为该词汇在文档中出现的频率。

3.2.3 文本长度规范化

语料中每个文本的长度是不一样的，文本长度对于文本表示也会产生影响。举一个极端的例子，将一段文本进行两倍长度的复制扩展，并使用上面提到的 TF 权重进行文

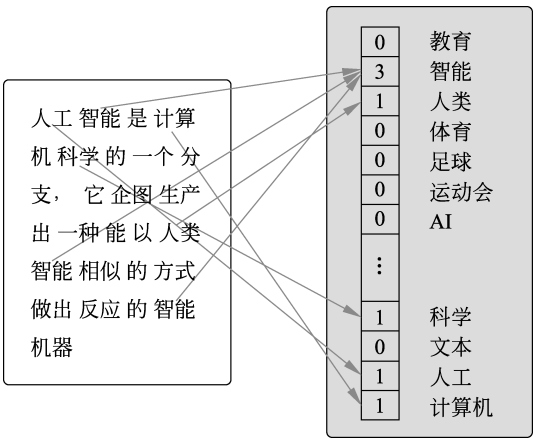


图 3.2 基于特征频率的特征权重

本表示，尽管扩展后的文本在信息量上并没有得到增加，但新的文本向量变成了原来的两倍。

因此，为了消除或减少文本长度对于文本表示的影响，需要对特征向量进行规范化处理，这一过程也称为文本长度归一化。对于文本 $\boldsymbol{d} = (w_1, w_2, \cdots, w_n)$，常见的长度规范化处理方法包括：

- 1-范数规范化

$$\boldsymbol{d}_1 = \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|_1} = \frac{\boldsymbol{d}}{\sum_i w_i} \tag{3.6}$$

规范后的向量落在 $w_1 + w_2 + \cdots + w_n = 1$ 的超平面上。

- 2-范数规范化

$$\boldsymbol{d}_2 = \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|_2} = \frac{\boldsymbol{d}}{\sqrt[2]{\sum_i w_i^2}} \tag{3.7}$$

规范后的向量落在 $w_1^2 + w_2^2 + \cdots + w_n^2 = 1$ 的球面上。

- 最大词频规范化

$$\boldsymbol{d}_{\max} = \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|_\infty} = \frac{\boldsymbol{d}}{\max\limits_i\{w_i\}} \tag{3.8}$$

需要说明的是，与机器学习和模式识别任务中常见的针对特征的去量纲归一化处理不同，文本表示中的归一化是针对样本的去长度因素进行的处理。

3.2.4 特征工程

向量空间模型假设空间中的坐标是两两正交的，即构成文本的特征项是相互独立的，与位置或顺序无关。事实上，这样的假设丢失了原始文档的词序、句法和部分语义信息等，虽然在一部分简单的文本挖掘任务上（如文本主题分类）这样的假设往往还算合

理，但是对于很多其他相对复杂的文本挖掘任务（如情感分析和观点挖掘），表现往往差强人意。比如，语义倾向性完全相反的两个文本“John is quicker than Mary”和“Mary is quicker than John”在词袋模型下的文本表示完全一致，这显然是不合理的。

因此，依据任务的要求，除了词以外，还可以将特征项定义为关键词、词组、短语等，并向词和词组中加入位置、词性、句法结构、语义等其他信息。在文本挖掘任务中，这种在向量空间模型中加入更多语言学或其他类型特征的做法称为特征工程（feature engineering）。

常用的语言学特征包括如下几种。

（1）n 元语法特征

基本的向量空间模型通常以词作为特征项，这种方法丢失了词序信息。n 元语法（n-gram）以词组（词序列）特征作为基本单元，可以捕捉一部分词序信息。以句子“我 强烈 推荐 这部 电影”为例，其一元语法、二元语法和三元语法特征见表 3.1。

表 3.1 n 元语法示例

语法模型	我 强烈 推荐 这部 电影
一元语法（unigram）	{我，强烈，推荐，这部，电影}
二元语法（bi-gram）	{我 强烈，强烈 推荐，推荐 这部，这部 电影}
三元语法（tri-gram）	{我 强烈 推荐，强烈 推荐 这部，推荐 这部 电影}

其中，一元语法特征即词项特征。n 元语法特征表示法在文本分类、文本聚类等领域得到了较为广泛的运用。但是，n 元语法并不是一种性价比较高的特征项，随着 n 的增大，特征空间的维数呈指数级增长，特征向量变得愈加稀疏，牺牲了统计质量，也增加了计算开销。同时，虽然 n 元语法能够体现邻接词组的关系，但是它难以捕捉句子中距离较远的词和词之间的关系。要捕捉这种关系信息，就要借助于更深层次的语言处理技术。

（2）句法特征

句法分析是自然语言处理的重要手段之一，其基本任务是确定句子的句法结构。它能够提供句子的句法信息，为后续的自然语言处理任务提供帮助。其中，依存关系分析是句法分析的一个重要分支，它用词和词之间的依存关系描述语言结构（宗成庆, 2013）。作为一种结构化的文本表示，一棵依存关系树以词为节点，用节点之间的指向关系表述词之间的支配和被支配关系。上述示例“我强烈推荐这部电影”的依存关系树如图 3.3 所示。

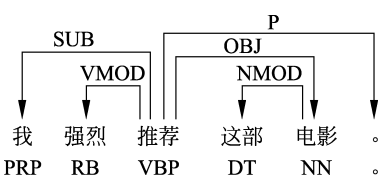


图 3.3 依存关系树示例

在向量空间模型中，一种简单的依存关系特征抽取方法是抽取相互依存的词对作为特征项，例如，上述例句中的“推荐电影”。这样一来，“推荐”和“电影”这种远距离依存关系就可以捕捉到了。

（3）语义知识库特征

一词多义、一义多词是自然语言中普遍存在的现象。识别两个单词是否表达同一个含义，或者判别多义词在文档中的具体含义，对于自然语言处理来说十分重要。借助额外的语义知识库（semantic knowledge base）（如英文的 WordNet、中文的知网 HowNet 等），利用词语在知识库中定义的语义概念等信息，作为词语的替代或者补充，可以在一定程度上解决歧义性和多样性给特征向量带来的噪声问题，提高文本表示的性能。

3.2.5 其他文本表示方法

除了用传统的向量空间模型表示文本和实现特征工程以外，还有一类分布式文本表示方法。与高维稀疏的向量空间模型不同，分布式表示方法通过建立主题模型或表示学习模型，实现文本的低维稠密表示。其代表性的方法包括文本概念表示和文本深度表示。

（1）文本概念表示

传统的向量空间模型是一种显式的文本表示方法，无法深入捕获文本中隐含的语义关系。以潜在语义分析（latent semantic analysis, LSA）、概率潜在语义分析（probabilistic latent semantic analysis, PLSA）和潜在狄利克雷分布（latent Dirichlet allocation LDA）为代表的主题模型，旨在挖掘文本中隐含的主题（topic）或概念（concept），可以较好地捕获多义性（polysemy）和同义性（synonymy），从而部分地解决一词多义和一义多词问题。同时，主题提供了一种高维文本数据维数的约减方法，将传统的向量空间模型中的高维稀疏向量转化为低维稠密向量，以缓解维数灾难问题，为文本表示提供了一种新的思路。本书将在第 7 章专门介绍文本主题模型和基于主题模型的文本概念表示。

（2）文本深度表示

文本表示学习的目标是通过机器学习方法，学习得到文本不同粒度单元的低维稠密向量。近年来，随着计算机计算能力的提升，基于人工神经网络（artificial neural network）的深度学习方法在自然语言处理中获得了很大的成功，涌现出了一系列基于深度学习的文本分布式表示方法。与传统的向量空间表示方法相比，分布式表示的向量维度较低，可有效缓解数据稀疏问题，从而提高计算效率。同时，表示学习方法在构造文本表示的过程中，可充分捕捉文本对象的语义信息和其他深度信息，避免了传统向量空间模型所需的复杂特征工程，在诸多文本挖掘任务中取得了高效的性能。在本书后面的章节里将对文本深度表示方法及其在不同文本挖掘任务中的应用方法分别进行介绍。

另外需要说明的是，文本表示的目的是构造适合自然语言处理任务的文本表示形式。对于不同的任务，文本表示的侧重点也有所不同。如针对文本情感分类任务的文本表示，需要在向量空间构造或表示学习过程中体现较多的文本情感属性，而在面向话题

检测和跟踪任务中的文本表示则需更多地体现事件描述信息等。因此，文本表示往往是任务相关的。面向不同的任务，不存在一种好而全的文本表示方法。在评价文本表示方法的优劣时，也需要结合不同任务的特点分别进行，酌情而定。

在文本分类和情感分析等文本数据挖掘任务中，词袋模型是最流行的文本表示方法。正如前面所述，词袋模型将每个文本视为一个词语的集合，集合的大小由所有文本统计出的词表规模决定，集合中的每个元素表示某个特定词语是否在当前文本中出现，或者表示该特定词语在当前文本中的统计权重。可见，是否出现或者出现的权重都是依据词语本身的字符串匹配统计得出的，因此，词语本身的离散符号表示是词袋模型的基础。而词语的离散符号表示等价于词的独热表示（one-hot representation），即每个词语利用一个布尔向量表示，向量维度为词表规模，其中只有当前词语对应的位置处为 1，其余位置都是 0。例如，假设文本分类的训练样本中统计出 5 万个不同的词汇，那么 5 万就是词表的规模。我们可以依据词语在训练样本中的出现顺序为所有词语进行编号，例如，词汇“文本”出现在第一个位置上，“挖掘”出现在最后一个位置上，那么“文本”和“挖掘”两个词汇的编号分别是 1 和 50000。每个词语对应唯一的编号，那么，一个词语就对应一个 5 万维的向量。例如，“文本”对应 [1，0，0，$\cdots$，0]，即除了第一个位置为 1，其余的 49999 个位置都是 0。

这种表示方法存在两个潜在的问题：一是基于 0 和 1 的离散符号匹配方法容易产生数据稀疏问题；二是任意两个词语在独热表示方法中都是相互独立的，即无法捕捉词语之间的语义相似性。近年来，在低维连续的语义向量空间中学习文本的分布式表示逐渐成为研究热点，并在情感分析和标题生成等文本挖掘任务中超越传统的词袋模型，取得了当前最佳的性能。下面我们将从词语、短语、句子到文档，分别介绍分布式表示的学习方法。

3.3 词的分布式表示

词是具有独立含义的最小的语言单位，是短语、句子和文档的基本组成单元。传统的独热表示方法无法刻画词语的语法和语义信息，那么，如何将语法和语义信息编码在词语的表示中，成为研究者关注的重点。Harris 和 Firth 分别于 1954 年和 1957 年提出并明确了词语的分布式假说：一个词的语义由其上下文决定，即上下文相似的词语，其语义也相似（Harris, 1954；Firth, 1957）。顾名思义，如果掌握了一个词所有的上下文信息，那么也就掌握了这个词的语义。因此，语料资源越丰富，获得的分布式表示越能够刻画词的语义信息。20 世纪 90 年代以来，随着统计方法的逐渐兴起和语料规模的快速扩大，如何学习词的分布式表示问题受到了越来越多的关注。简单地说，分布式表示学习的核心思想就是利用低维连续的实数向量表示一个词语，使得语义相近的词在实数向量空间中也临近。本节着重介绍几种典型的词的分布式表示方法。

分布式假说表明词语表示的质量很大程度上取决于对上下文信息的建模。在基于矩阵分解的分布式表示方法中，最常用的上下文是固定窗口中的词语集合，很难利用更加复杂的上下文信息。例如，若采用窗口内的 n 元语法（n-gram）作为上下文，n-gram 数

目将会随着 n 的增加呈指数级增长，数据稀疏和维数灾难问题将不可避免。神经网络模型实质上是由一系列线性组合和非线性变换等简单操作构成，理论上可以模拟任意函数，因此，可以对复杂的上下文通过简单的神经网络结构进行建模，从而使得词语的分布式表示能够捕捉更多的句法和语义信息。

不同于矩阵分解方法中的文档集合表示，神经网络模型中的训练数据都以句子集合的形式表示：$D = \{w_{i1}^{m_i}\}_{i=1}^{M}$，其中，$m_i$ 表示第 i 个句子包含的词语数目，$w_{i1}^{m_i}$ 表示该句子的词序列 $w_{i1}\ w_{i2}\ \cdots\ w_{mi}$。统计训练数据集 D 中出现的词语，可以得到一个词汇表 V[①]，假设每个词语映射到一个 d 维[②]的分布式向量（通常称为词向量），那么词汇表 V 对应一个词向量矩阵 $\boldsymbol{L} \in \mathbb{R}^{|V| \times d}$。神经网络模型的目标在于如何优化词向量矩阵 $\boldsymbol{L}$，为每个词语学习准确的分布式向量表示。以下介绍几种常用的神经网络模型。

3.3.1 神经网络语言模型

在神经网络模型中，词向量表示最初用于神经网络语言模型的学习过程。语言模型用来计算一段文本的出现概率，度量该文本的流畅程度。给定 m 个词语构成的句子 $w_1\ w_2\ \cdots\ w_m$，其出现的可能性可通过链式规则计算：

$$p(w_1 w_2 \cdots w_m) = p(w_1)\, p(w_2|w_1) \cdots p(w_i|w_1, w_2, \cdots, w_{i-1}) \cdots p(w_m|w_1, w_2, \cdots, w_{m-1}) \tag{3.9}$$

在传统语言模型建模过程中，通常基于相对频率的最大似然估计（maximum likelihood estimation, MLE）方法估计条件概率 $p(w_i|w_1, w_2, \cdots, w_{i-1})$：

$$p(w_i|w_1, w_2, \cdots, w_{i-1}) = \frac{\text{count}(w_1, w_2, \cdots, w_i)}{\text{count}(w_1, w_2, \cdots, w_{i-1})} \tag{3.10}$$

由于 i 越大，词组 $w_1, w_2, \cdots, w_i$ 出现的可能性越小，最大似然估计越不准确。因此，典型的解决方案是采用 $(n-1)$ 阶马尔可夫链对语言模型进行建模（即 n 元语言模型），假设当前词的出现概率仅依赖于前 $(n-1)$ 个词：

$$p(w_i|w_1, w_2, \cdots, w_{i-1}) \approx p(w_i|w_{i-n+1}, \cdots, w_{i-1}) \tag{3.11}$$

若 $n = 1$，表示一元语言模型（unigram），假设词语之间是相互独立的；$n = 2$ 表示二元语言模型（bigram），当前词的出现概率与前一个词有关。$n = 3$、$n = 4$ 和 $n = 5$ 是使用最广泛的几种 n 元语言模型。这种近似方法使得词序列的语言模型概率计算成为可能。但是，基于词、词组等离散符号匹配的概率估计方法仍然面临严重的数据稀疏问题，并且无法捕捉词语之间的语义相似性。例如，两个二元词组“很无聊”和“很枯燥”的语义非常相近，p（无聊 | 很）与 p（枯燥 | 很）的概率应该非常接近，但实际上这两个二元词组在数据中的频率可能差别悬殊，导致两个概率 p（无聊 | 很）与 p（枯燥 | 很）的差别也较大。

Bengio 等提出了一种基于前馈神经网络（feed-forward neural network, FNN）的语言模型（Bengio et al., 2003），其基本思路是：将每个词映射为一个低维连续的实数向

① 一般根据词频对词汇表进行限制，例如，保留词频大于某个阈值的所有词语，或者保留词频最高的 $|V|$ 个词。

② d 是一个经验值，与具体应用有关，一般选取几十、几百或者一千左右。

量（即词向量），并在连续向量空间中对 n 元语言模型的概率 $p(w_i|w_{i-n+1},\cdots,w_{i-1})$ 进行建模。图 3.4（a）展示了一个三层的前馈神经网络语言模型。首先，历史信息的 $(n-1)$ 个词被映射为词向量，并被拼接后得到 $\boldsymbol{h}_0$：

$$\boldsymbol{h}_0 = [\boldsymbol{e}(w_{i-n+1});\cdots;\boldsymbol{e}(w_{i-1})] \tag{3.12}$$

其中，$\boldsymbol{e}(w_{i-1}) \in \mathbb{R}^d$ 表示词语 w_{i-1} 对应的 d 维词向量，可通过检索词向量矩阵[①] $\boldsymbol{L} \in \mathbb{R}^{|V|\times d}$ 获得。$\boldsymbol{h}_0$ 通过非线性隐藏层学习 $(n-1)$ 个词的抽象表示：

$$\boldsymbol{h}_1 = f\left(\boldsymbol{U}^1 \times \boldsymbol{h}_0 + \boldsymbol{b}^1\right) \tag{3.13}$$

$$\boldsymbol{h}_2 = f\left(\boldsymbol{U}^2 \times \boldsymbol{h}_1 + \boldsymbol{b}^2\right) \tag{3.14}$$

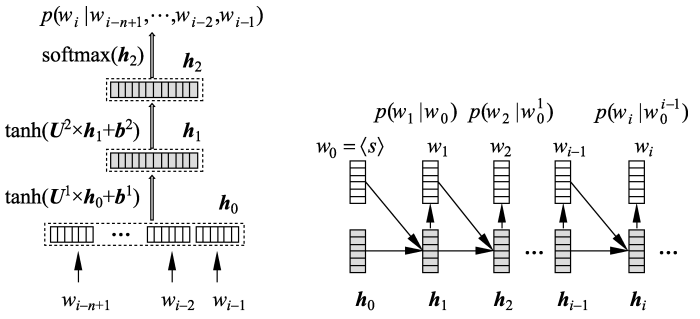
其中，非线性激活函数可选择 $f(\cdot) = \tanh(\cdot)$。最后，利用 softmax 函数计算词表 V 中每个词的概率分布：

$$p(w_i|w_{i-n+1},\cdots,w_{i-1}) = \frac{\exp(\boldsymbol{h}_2 \cdot \boldsymbol{e}(w_i))}{\sum_{k=1}^{|V|}\exp(\boldsymbol{h}_2 \cdot \boldsymbol{e}(w_k))} \tag{3.15}$$

上述公式中，权重矩阵 $\boldsymbol{U}^1$，$\boldsymbol{U}^2$，$\boldsymbol{b}^1$，$\boldsymbol{b}^2$ 和词向量矩阵 $\boldsymbol{L}$ 都视为神经网络的参数 θ。训练过程便是优化参数 θ，使得整个训练数据上的对数似然值最大：

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{m=1}^{M} \log p(w_{i1}^{m_i}) \tag{3.16}$$

语言模型训练结束后，就得到了优化后的词向量矩阵 $\boldsymbol{L}^*$，它包含了词表 V 中所有词语的分布式向量表示。



(a) 前馈神经网络语言模型　　(b) 循环神经网络语言模型

图 3.4　神经网络语言模型示意图

由于前馈神经网络语言模型仅能对固定窗口的上下文进行建模，无法捕捉长距离的上下文依赖关系，Mikolov 等便提出了采用循环神经网络（recurrent neural network, RNN） 直接对概率 $p(w_i|w_1,w_2,\cdots,w_{i-1})$ 进行建模的思路（Mikolov et al., 2010），旨

① 词向量矩阵 $\boldsymbol{L}$ 一般可随机地进行初始化，并在模型训练过程中作为参数进行优化。

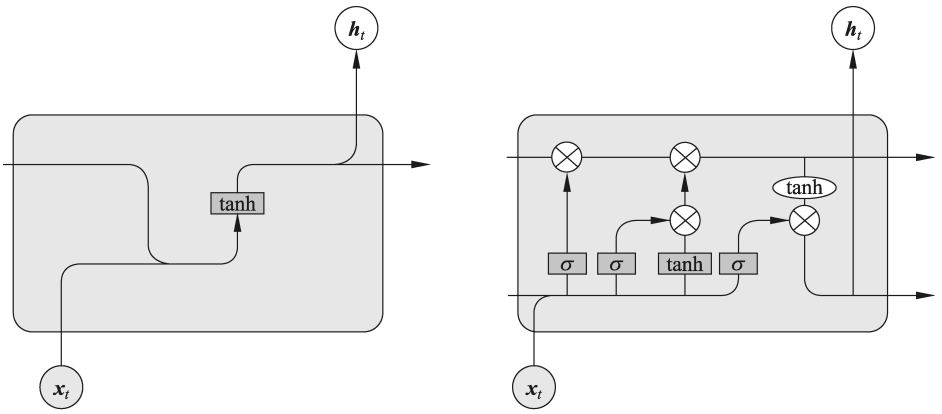
在利用所有的历史信息 $w_1, w_2, \cdots, w_{i-1}$ 预测当前词 w_i 的出现概率。循环神经网络的核心要点在于计算每一时刻的隐藏层表示 $\boldsymbol{h}_i$：

$$\boldsymbol{h}_i = f\left(\boldsymbol{W}\boldsymbol{e}\left(w_{i-1}\right) + \boldsymbol{U}\boldsymbol{h}_{i-1} + \boldsymbol{b}\right) \tag{3.17}$$

其中，第 $i-1 (i \geqslant 2)$ 时刻的隐藏层表示 $\boldsymbol{h}_{i-1}$ 蕴含从第 0 时刻到 $(i-1)$ 时刻的历史信息（第 0 时刻的历史信息通常设置为空，即 $\boldsymbol{h}_0 = 0$）。$f(\cdot)$ 为非线性激活函数，可取 $f(\cdot) = \tanh(\cdot)$。在第 i 时刻隐藏层表示 $\boldsymbol{h}_i$ 的基础上，可直接采用 softmax 函数计算下一个词 w_i 的出现概率 $p(w_i|w_1, w_2, \cdots, w_{i-1})$。神经网络参数和词向量矩阵的优化方法与前馈神经网络方法类似，都是最大化训练数据的对数似然。

针对 RNN 在处理长序列输入时容易出现梯度消失或梯度爆炸的问题，(Hochreiter and Schmidhuber, 1997) 提出了长短时记忆（long-short term memory, LSTM）模型。

图 3.5 比较了传统的 RNN 和 LSTM 模型结构。在图 3.5(a) 所示的 RNN 中，当前时刻的输入与上一时刻的隐层状态进行线性变化并相加，再经过非线性激活后得到当前时刻的隐层状态。在 RNN 的基础上，图 3.5(b) 所示的 LSTM 增加了单元状态或称细胞状态（cell state）和三个门控机制：输入门 $\boldsymbol{i}_t$、遗忘门 $\boldsymbol{f}_t$ 和输出门 $\boldsymbol{o}_t$。其核心是单元状态，它作为整个模型的记忆空间，可以被理解为一种传送带，随着时间变化传送模型的记忆信息。传送带的记忆控制通过三个控制门实现。



(a) RNN (b) LSTM

图 3.5 LSTM 与传统 RNN 模型的比较

三个控制门将当前时刻的输入 $\boldsymbol{x}_t$、上一时刻的隐层状态 $\boldsymbol{h}_{t-1}$ 和单元状态 $\boldsymbol{c}_{t-1}$ 的线性变化相加，再用 sigmoid 函数激活，得到一个 $[0, 1]$ 之间的门限作为输出：

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{\mathrm{i}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{i}}\boldsymbol{h}_{t-1} + \boldsymbol{V}_{\mathrm{i}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathrm{i}}) \tag{3.18}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{\mathrm{f}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{f}}\boldsymbol{h}_{t-1} + \boldsymbol{V}_{\mathrm{f}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathrm{f}}) \tag{3.19}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{\mathrm{o}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{o}}\boldsymbol{h}_{t-1} + \boldsymbol{V}_{\mathrm{o}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathrm{o}}) \tag{3.20}$$

其中，σ 表示 sigmoid 函数，$\boldsymbol{W}_{\mathrm{i}}, \boldsymbol{V}_{\mathrm{i}}, \boldsymbol{b}_{\mathrm{i}}, \boldsymbol{W}_{\mathrm{f}}, \boldsymbol{V}_{\mathrm{f}}, \boldsymbol{b}_{\mathrm{f}}, \boldsymbol{W}_{\mathrm{o}}, \boldsymbol{V}_{\mathrm{o}}, \boldsymbol{b}_{\mathrm{o}}$ 分别为输入门、遗忘门和输出门的参数。

假设 $\boldsymbol{c}_{t-1}$ 为上一时刻的单元状态，$\tilde{\boldsymbol{c}}_t$ 是当前时刻的候选状态，那么，

$$\tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{\mathrm{c}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{c}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{c}}) \tag{3.21}$$

上述门限与状态或输入进行点乘，决定传送带上多少信息可以被传送过去：当控制门的输出值为 0 时不传送；当输出值为 1 时全部传送。例如，输入门与候选状态点乘，可以控制将多少当前时刻的状态信息输入到传送带，而遗忘门与上一时刻的单元状态点乘，则控制需要遗忘多少过去时刻的状态信息。两者相加得到当前时刻的单元状态：

$$\boldsymbol{c}_t = \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t + \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} \tag{3.22}$$

其中，$\odot$ 表示向量的点乘操作（element-wise multiplication）。

最后，单元状态经 tanh 非线性激活后与输出门点乘，得到当前时刻的隐层状态：

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \tag{3.23}$$

在标准的 RNN 模型中，每个词的表示只受位置之前的词的影响，位置之后的词对其不产生影响。为了更好地利用前向和后向的上下文信息，（Schuster and Paliwal, 1997）提出了双向 RNN（bi-directional RNN）模型。（Graves et al., 2013）在语音识别任务中使用了双向 LSTM（bi-LSTM），分别从前向后和从后向前两个方向对序列单元进行编码表示：

$$\overrightarrow{\boldsymbol{c}}_t, \overrightarrow{\boldsymbol{h}}_t = \mathrm{LSTM}(\overrightarrow{\boldsymbol{c}}_{t-1}, \overrightarrow{\boldsymbol{h}}_{t-1}, \boldsymbol{x}_t) \tag{3.24}$$

$$\overleftarrow{\boldsymbol{c}}_t, \overleftarrow{\boldsymbol{h}}_t = \mathrm{LSTM}(\overleftarrow{\boldsymbol{c}}_{t+1}, \overleftarrow{\boldsymbol{h}}_{t+1}, \boldsymbol{x}_t) \tag{3.25}$$

并将前向 LSTM 得到的隐层状态 $\overrightarrow{\boldsymbol{h}}_t$ 与后向 LSTM 得到的隐层状态 $\overleftarrow{\boldsymbol{h}}_t$ 拼接起来，作为最终的隐层状态：

$$\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{h}}_t] \tag{3.26}$$

模型结构如图 3.6 所示。这种双向结构同样适用其他 RNN 模型。

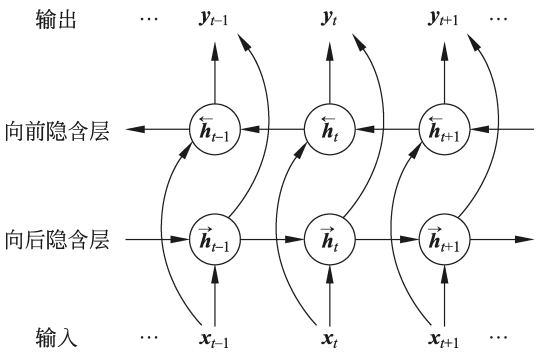


图 3.6 Bi-LSTM 结构示意图

针对 LSTM 门控网络结构复杂和存在冗余的缺点，论文（Cho et al., 2014）在 LSTM 的基础上提出了一种名为门控循环单元（gated recurrent unit, GRU）的 LSTM 变体。GRU 将遗忘门和输入门合并为更新门，同时将单元状态和隐藏层进行合并，从而简化了 LSTM 模型的结构。如图 3.7 所示。

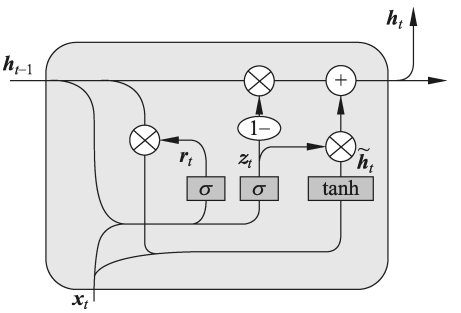


图 3.7 GRU 网络结构图

GRU 主要包含两个门控模块：重置门（reset gate）和更新门（update gate）。重置门主要决定有多少过去的信息需要遗忘，更新门则主要用于决定将多少过去的信息传递到未来：

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_{\mathrm{r}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{r}}) \tag{3.27}$$

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_{\mathrm{z}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{z}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{z}}) \tag{3.28}$$

基于重置门计算当前时刻的候选状态：

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_{\mathrm{h}}\boldsymbol{x}_t + \boldsymbol{U}_{\mathrm{h}}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_{\mathrm{h}}) \tag{3.29}$$

并基于更新门对隐层状态进行更新：

$$\boldsymbol{h}_t = \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t \tag{3.30}$$

其中，$\sigma(\cdot)$ 是 sigmoid 函数，$\boldsymbol{W}_{\mathrm{r}}, \boldsymbol{W}_{\mathrm{h}}, \boldsymbol{W}_{\mathrm{z}}, \boldsymbol{U}_{\mathrm{r}}, \boldsymbol{U}_{\mathrm{h}}, \boldsymbol{U}_{\mathrm{z}}$ 是参数矩阵。

GRU 在结构上比 LSTM 简单，参数更少，但在实践中与 LSTM 相比性能没有明显的劣势，甚至在一些任务上效果更好，因此也成为一种较为流行的 RNN 模型。

LSTM，GRU 和 Bi-LSTM 可以有效地捕捉长距离的语义依赖关系，在文本摘要和信息抽取等很多序列预测的文本挖掘任务中都体现出更优的性能（Nallapati et al., 2016; See et al., 2017）。

3.3.2 C&W 模型

在神经网络语言模型中，词向量的表示学习只是一个副产品，并不是核心任务。Collobert 和 Weston 于 2008 年提出了一种模型，直接以学习和优化词向量为最终

目标，这种模型以两位学者的姓氏首字母命名，称为 C&W 模型（Collobert and Weston model）（Collobert and Weston, 2008）。

神经网络语言模型的目标在于准确估计条件概率 $p(w_i|w_1, w_2, \cdots, w_{i-1})$，因此每一时刻都需要利用隐藏层到输出层的矩阵运算和 softmax 函数计算整个词汇表的概率分布，计算复杂度为 $O(|\boldsymbol{h}| \times |V|)$，其中 $|\boldsymbol{h}|$ 是最高隐藏层的神经元数目（通常为几百或一千左右），$|V|$ 是词表规模（通常为几万至十万左右）。这个矩阵运算操作极大地降低了模型的训练效率。Collobert 和 Weston 认为，如果目标只是学习词向量，则没有必要采用语言模型的方式，而可以直接从分布式假说的角度设计模型和目标函数：给定训练语料中任意一个 n 元组（$n = 2C + 1$）$(w_i, C) = w_{i-C} \cdots w_{i-1} w_i w_{i+1} \cdots w_{i+C}$，如果将中心词 w_i 随机地替换为词表中的其他词 w_i'，得到一个新的 n 元组 $(w_i', C) = w_{i-C} \cdots w_{i-1} w_i' w_{i+1} \cdots w_{i+C}$，那么，$(w_i, C)$ 一定比 (w_i', C) 更加合理。如果对每个 n 元组进行打分，那么 (w_i, C) 得分一定比 (w_i', C) 高，即

$$s(w_i, C) > s(w_i', C) \tag{3.31}$$

如图 3.8 所示，简单的前馈神经网络模型只需要计算 n 元组的得分，并从得分能够区分输入的 n 元组是来自于真实的（right）训练文本，还是随机生成的（random）文本。我们将真实训练文本中的 n 元组 (w_i, C) 称为正样本，随机生成的 n 元组 (w_i', C) 称为负样本。

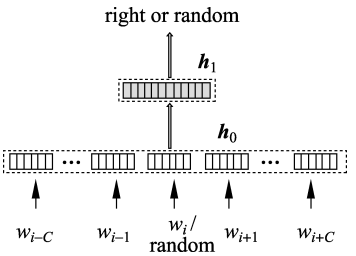


图 3.8 C&W 模型示意图

为了计算 $s(w_i, C)$，首先将 $w_{i-C} \cdots w_{i-1}\ w_i\ w_{i+1} \cdots w_{i+C}$ 中的每个词从词向量矩阵 $\boldsymbol{L}$ 中获得对应的词向量，并进行拼接，得到第一层表示 $\boldsymbol{h}_0$：

$$\boldsymbol{h}_0 = [\boldsymbol{e}(w_{i-C}); \cdots; \boldsymbol{e}(w_{i-1}); \boldsymbol{e}(w_i); \boldsymbol{e}(w_{i+1}); \cdots; \boldsymbol{e}(w_{i+C})] \tag{3.32}$$

$\boldsymbol{h}_0$ 经过隐藏层得到 $\boldsymbol{h}_1$：

$$\boldsymbol{h}_1 = f(\boldsymbol{W}_0 \boldsymbol{h}_0 + \boldsymbol{b}_0) \tag{3.33}$$

其中，$f(\cdot)$ 为非线性激活函数。$\boldsymbol{h}_1$ 再经过线性变换，得到 n 元组 (w_i, C) 的得分：

$$s(w_i, C) = \boldsymbol{W}_1 \boldsymbol{h}_1 + \boldsymbol{b}_1 \tag{3.34}$$

其中，$\boldsymbol{W}_1 \in \mathbb{R}^{1 \times |\boldsymbol{h}_1|}$，$\boldsymbol{b}_1 \in \mathbb{R}^1$。可见 C&W 模型由隐藏层到输出层的矩阵运算非常简单，将计算复杂度由神经网络语言模型的 $O(|\boldsymbol{h}| \times |V|)$ 降低至 $O(|\boldsymbol{h}|)$，可以高效地学习词向量表示。

在词向量优化过程中，C&W 模型希望每一个正样本的打分比对应负样本的打分高 1 分，即

$$s(w_i, C) > s(w_i', C) + 1 \tag{3.35}$$

对于整个训练语料，C&W 模型需要遍历语料中的每个 n 元组，并最小化如下的目标函数：

$$\sum_{(w_i,C)\in D} \sum_{w_i'\in V_{\text{neg}}} \max(0, 1 + s(w_i', C) - s(w_i, C)) \tag{3.36}$$

其中，V_{neg} 为负样本集合。

3.3.3 CBOW 与 Skip-gram 模型

无论采用神经网络语言模型还是 C&W 模型，隐藏层都是不可或缺的，而输入层到隐藏层的矩阵运算也是高额时间开销的关键部分。为了进一步简化神经网络结构，更加高效地学习词向量表示，Mikolov 等在 2013 年提出了两种不含隐藏层的神经网络模型：CBOW 模型（continuous bag-of-words model）和 Skip-gram 模型（Skip-gram model）（Mikolov et al., 2013a）。

1. CBOW 模型

如图 3.9 所示，CBOW 模型的思想类似于 C&W 模型：输入上下文词语，预测中心目标词语。不同于 C&W 模型，CBOW 模型仍然以目标词的概率为优化目标，而且 CBOW 模型在网络结构设计上做了两点简化：一方面，输入层不再是上下文词对应词向量的拼接，而是忽略词序信息，直接采用所有词向量的平均值；另一方面，省略隐藏层，输入层直接与输出层连接，采用 logistic 回归（logistic regression）的形式计算中心目标词的概率。

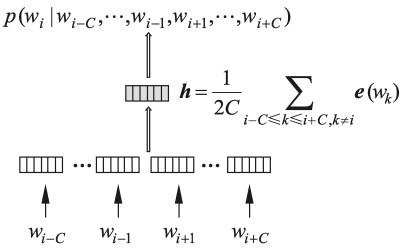


图 3.9 CBOW 模型示意图

形式化地，给定训练语料中任意一个 n 元组（$n = 2C + 1$）$(w_i, C) = w_{i-C}\cdots w_{i-1}w_i w_{i+1}\cdots w_{i+C}$，将 $WC = w_{i-C}\cdots w_{i-1}w_{i+1}\cdots w_{i+C}$ 作为输入，计算上下文词的平均词向量：

$$\boldsymbol{h} = \frac{1}{2C} \sum_{i-C\leqslant k\leqslant i+C,k\neq i} \boldsymbol{e}(w_k) \tag{3.37}$$

$\boldsymbol{h}$ 直接作为上下文的语义表示预测中心目标词 w_i 的概率：

$$p\left(w_i|WC\right)=\frac{\exp\left(\boldsymbol{h}\cdot\boldsymbol{e}\left(w_i\right)\right)}{\sum_{k=1}^{|V|}\exp\left(\boldsymbol{h}\cdot\boldsymbol{e}\left(w_k\right)\right)} \tag{3.38}$$

在 CBOW 模型中，词向量 $\boldsymbol{L}$ 是唯一的神经网络参数。对于整个训练语料，CBOW 模型优化词向量矩阵 $\boldsymbol{L}$ 以最大化所有词的对数似然：

$$\boldsymbol{L}^*=\underset{\boldsymbol{L}}{\operatorname{argmax}}\sum_{w_i\in V}\log p\left(w_i|WC\right)$$

2. Skip-gram 模型

与 CBOW 模型利用上下文词预测中心词的做法不同，Skip-gram 模型采用了相反的过程，即用中心词预测所有上下文词。图 3.10 展示了 Skip-gram 模型的基本思想。

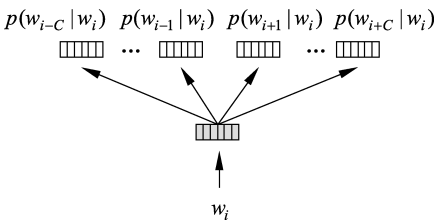


图 3.10 Skip-gram 模型示意图

给定训练语料中任意一个 n 元组 $(w_i,C)=w_{i-C}\cdots w_{i-1}w_iw_{i+1}\cdots w_{i+C}$，Skip-gram 模型直接利用中心词 w_i 的词向量 $\boldsymbol{e}\left(w_i\right)$ 预测上下文 $WC=w_{i-C}\cdots w_{i-1}w_{i+1}\cdots w_{i+C}$ 中每个词 w_c 的概率：

$$p\left(w_c|w_i\right)=\frac{\exp\left(\boldsymbol{e}\left(w_i\right)\cdot\boldsymbol{e}\left(w_c\right)\right)}{\sum_{k=1}^{|V|}\exp\left(\boldsymbol{e}\left(w_i\right)\cdot\boldsymbol{e}\left(w_k\right)\right)} \tag{3.39}$$

Skip-gram 模型的目标函数与 CBOW 模型的目标函数类似，都是优化词向量矩阵 $\boldsymbol{L}$ 以最大化所有上下文词的对数似然：

$$\boldsymbol{L}^*=\underset{\boldsymbol{L}}{\operatorname{argmax}}\sum_{w_i\in V}\sum_{w_c\in WC}\log p\left(w_c|w_i\right) \tag{3.40}$$

3.3.4 噪声对比估计与负采样

CBOW 模型和 Skip-gram 模型虽然极大地简化了神经网络结构，但是仍然需要利用 softmax 函数计算词汇表 V 中所有词的概率分布。为了加速神经网络模型的训练效

率，Mikolov 等受 C&W 模型和噪声对比估计（noise contrastive estimation, NCE）方法的启发，提出了负采样（negative sampling, NEG）技术（Mikolov et al., 2013b）。

以 Skip-gram 模型为例，通过中心词 w_i 预测上下文 $WC=w_{i-C}\cdots w_{i-1}w_{i+1}\cdots w_{i+C}$ 中的任意词 w_c，负采样技术和噪声对比估计方法都是为每个正样本 w_c 从某个概率分布 $p_n(w)$ 中选择 K 个负样本 $\{w'_1, w'_2, \cdots, w'_K\}$，并最大化正样本的似然，同时最小化所有负样本的似然。

对于一个正样本 w_c 和 K 个负样本 $\{w'_1, w'_2, \cdots, w'_K\}$，噪声对比估计方法首先对 $K+1$ 个样本的概率进行归一化：

$$\begin{aligned} p(l=1, w|w_i) &= p(l=1)\, p(w|l=1, w_i) \\ &= \frac{1}{K+1} p_\theta(w|w_i) \end{aligned} \tag{3.41}$$

$$\begin{aligned} p(l=0, w|w_i) &= p(l=0)\, p(w|l=0, w_i) \\ &= \frac{K}{K+1} p_n(w) \end{aligned} \tag{3.42}$$

$$\begin{aligned} p(l=1|w, w_i) &= \frac{p(l=1, w|w_i)}{p(l=0, w|w_i) + p(l=1, w|w_i)} \\ &= \frac{p_\theta(w|w_i)}{p_\theta(w|w_i) + K p_n(w)} \end{aligned} \tag{3.43}$$

$$\begin{aligned} p(l=0|w, w_i) &= \frac{p(l=0, w|w_i)}{p(l=0, w|w_i) + p(l=1, w|w_i)} \\ &= \frac{K p_n(w)}{p_\theta(w|w_i) + K p_n(w)} \end{aligned} \tag{3.44}$$

其中，w 表示某个样本；$l=1$ 表示该样本来自于正样本，服从神经网络模型的概率输出[①] $p_\theta(w|WC)$；$l=0$ 表示该样本来自于负样本，服从噪声样本生成的概率分布 $p_n(w)$。噪声对比估计的目标函数如下：

$$J(\theta) = \log p(l=1|w_c, w_i) + \sum_{k=1}^{K} \log p(l=0|w'_k, w_i) \tag{3.45}$$

负采样技术的目标函数与噪声对比估计相同，但不同于噪声对比估计方法的是负采样技术不对样本集合进行概率归一化，而直接采用神经网络语言模型输出：

$$p(l=1|w_c, w_i) = \frac{1}{1+\mathrm{e}^{-\boldsymbol{e}(w_i)\cdot\boldsymbol{e}(w_c)}} \tag{3.46}$$

那么，目标函数可以简化为

① $p_\theta(w|WC) = \dfrac{\exp(\boldsymbol{h}\cdot\boldsymbol{e}(w))}{\sum_{k=1}^{|V|}\exp(\boldsymbol{h}\cdot\boldsymbol{e}(w_k))} = \dfrac{\exp(\boldsymbol{h}\cdot\boldsymbol{e}(w_i))}{z(w)}$，在 NCE 方法中一般取 $z(w)=1$。

$$
\begin{aligned}
J(\theta) &= \log p(l=1|w_c, w_i) + \sum_{k=1}^{K} \log p(l=0|w'_k, w_i) \\
&= \log p(l=1|w_c, w_i) + \sum_{k=1}^{K} \log\left(1 - p(l=1|w'_k, w_i)\right) \\
&= \log \frac{1}{1+\mathrm{e}^{-\boldsymbol{e}(w_i)\cdot\boldsymbol{e}(w_c)}} + \sum_{k=1}^{K} \log\left(1 - \frac{1}{1+\mathrm{e}^{-\boldsymbol{e}(w'_k)\cdot\boldsymbol{e}(w_c)}}\right) \\
&= \log \frac{1}{1+\mathrm{e}^{-\boldsymbol{e}(w_i)\cdot\boldsymbol{e}(w_c)}} + \sum_{k=1}^{K} \log\left(\frac{1}{1+\mathrm{e}^{\boldsymbol{e}(w'_k)\cdot\boldsymbol{e}(w_c)}}\right) \\
&= \log \sigma(\boldsymbol{e}(w_i)\cdot\boldsymbol{e}(w_c)) + \sum_{k=1}^{K} \log \sigma(-\boldsymbol{e}(w'_k)\cdot\boldsymbol{e}(w_c))
\end{aligned} \tag{3.47}
$$

Mikolov 等通过实验发现，负样本数目 $K=5$ 时就能够取得很好的性能。可见，负采样技术极大地简化了概率估计方法，有效提升了词向量的学习效率。

3.3.5 字词混合的分布式表示方法

基于分布式假说的词向量表示学习需要足够的上下文信息去捕捉一个词的语义，也就是说，要求词出现的频率足够高。但是，根据齐夫定律（Zipf's Law），绝大多数词在语料中很少出现。对于这些词，无法依据分布式假说获得高质量的词向量表示。

虽然词是能够独立运用的最小语义单元，但是词并不是最小的语言单位，而是由字符或字构成的。例如，英文单词由字母组成，中文词由汉字构成。以中文词为例，研究者分析发现 93%的中文词满足或部分满足语义组合特性[①]，即这些词是语义透明的。如果一个词是语义透明的，表明这个词的语义可以由内部汉字的语义组合而成。如图 3.11 中的词“出租车”，“出”“租”“车”三个汉字的语义进行合成，便能得到“出租车”的语义。相比于词汇规模，汉字集合是有限的，根据国标 GB 2312 常用的汉字不足 7000 个，而且汉字在语料中的频率都比较高，能够在分布式假说下获得高质量的汉字向量。因此，如果能够充分挖掘汉字的语义向量表示，设计准确的语义组合函数，就能够极大地增强汉语词（特别是低频词）的向量表示能力。基于这种想法，字词混合的分布式表示方法越来越受到研究者的关注（Chen et al., 2015a；Xu et al., 2016；Wang et al., 2017a）。

字词混合的分布式表示方法可以有多种，它们之间的区别主要在于两方面：一是如何设计准确的汉字语义组合函数；二是如何融合汉字组合语义和中文词语的原子语义。下面以 C&W 模型的思想为例介绍两种字词混合的分布式表示方法。

所有方法的目标仍然是区分正常的 n 元组和随机的 n 元组，核心任务还是计算一个 n 元组的得分。图 3.11（a）是一种简单而直接的字词混合方法。假设中文词 $w_i = c_1c_2\cdots c_l$ 由 l 个汉字组成（例如，“出租车”由 3 个汉字组成），该方法首先学习汉字串 $c_1c_2\cdots c_l$ 的语义向量组合表示 $\boldsymbol{x}(c_1c_2\cdots c_l)$ 和中文词 w_i 的原子向量表示

① 其中，70%是部分满足，30%是完全满足。

$\boldsymbol{x}(w_i)$。在组合汉字的语义向量时，假设各个汉字的贡献相同，利用平均字向量表示 $\boldsymbol{x}(c_1c_2\cdots c_l)$：

$$\boldsymbol{x}(c_1c_2\cdots c_l)=\frac{1}{l}\sum_{k=1}^{l}\boldsymbol{x}(c_k) \tag{3.48}$$

其中，$\boldsymbol{x}(c_k)$ 表示汉字 c_k 的向量表示。为了获得最终的词向量，该方法直接将汉字的语义组合表示和中文词向量表示进行拼接：

$$\boldsymbol{X}_i=[\boldsymbol{x}(c_1c_2\cdots c_l);\boldsymbol{x}(w_i)] \tag{3.49}$$

之后的 $\boldsymbol{h}_0$，$\boldsymbol{h}_1$ 和最终得分的计算与 C&W 模型相同。

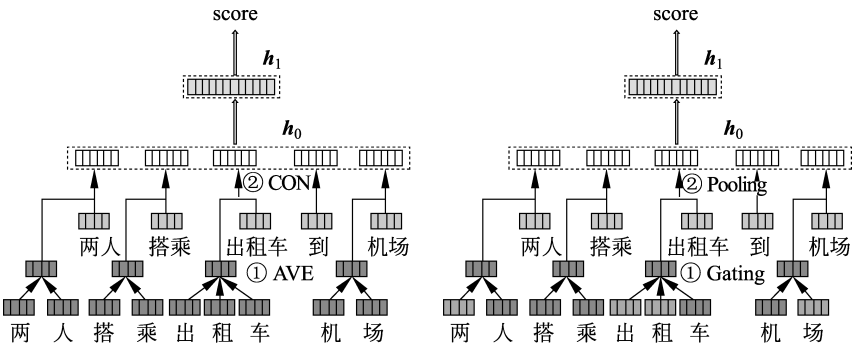


图 3.11 字词混合的词向量表示方法

不难看出，上述方法并未考虑不同的汉字对组合语义的影响，也没考虑组合语义和原子语义对最终词向量的影响。例如，在中文词语“出租车”中，汉字“车”的贡献最大，“出”和“租”仅起修饰作用，贡献相对较小。可见，不同汉字不应该等同视之。另一方面，有的词是透明词，更多地依赖组合语义，而有的词是非透明的（例如，“苗条”），则更多地依赖词的原子语义。图 3.11（b）所展示的是同时考虑上述两点因素的字词混合方法。首先通过门限（gating）机制获得汉字的组合语义：

$$\boldsymbol{x}(c_1c_2\cdots c_l)=\sum_{k=1}^{l}\boldsymbol{v}_k\odot\boldsymbol{x}(c_k) \tag{3.50}$$

其中，$\boldsymbol{v}_k\in\mathbb{R}^d$ 表示控制门，控制汉字 c_k 的向量 $\boldsymbol{x}(c_k)$ 对组合语义的贡献，可通过如下方式计算：

$$\boldsymbol{v}_k=\tanh(W[\boldsymbol{x}(c_k);\boldsymbol{x}(w_i)]) \tag{3.51}$$

其中，$\boldsymbol{W}\in\mathbb{R}^{d\times 2d}$。在融合组合语义和原子语义时，通过最大池化（max-pooling）方式获得：

$$\boldsymbol{X}_i=\max_{k=1}^{d}(\boldsymbol{x}(c_1c_2\cdots c_l)_k,\boldsymbol{x}(w_i)_k) \tag{3.52}$$

通过池化机制，可以学习出最终词的语义更加依赖于哪一种语义（是组合语义还是原子语义）。大量的实验表明，考虑词内汉字贡献度后获得的词向量具有更准确的表达能力。

3.4 短语的分布式表示

在统计自然语言处理中，短语一般指连续的词串，并非只是句法意义上的名词短语、动词短语和介词短语等。短语的分布式表示学习方法分为两种：一种方法视短语为不可分割的独立语义单元，然后基于分布式假说学习短语的语义向量表示；另一种方法认为短语的语义由词组合而成，关键是学习词和词之间的语义组合方式。

与词相比，短语的出现频率更低，因此基于分布式假说的短语向量表示在质量上无法得到保证。Mikolov 等只是将部分英语常见短语（例如，“New York Times”和“United Nations”等）视为不可分割的语义单元，与词等同对待（例如，“New_York_Times”和“United_Nations”），利用 CBOW 模型或 Skip-gram 模型学习相应的分布式表示。可见，这类方法无法适用于普通的短语表示学习。

3.4.1 基于词袋的分布式表示

基于组合语义的短语表示学习是一种更加自然合理的方法。如何将词的语义组合成短语的语义是这类表示学习方法的核心。给定一个由 i 个词组成的短语 $\mathrm{ph}_i = w_1 w_2 \cdots w_i$，最简单的语义组合方法就是采用词袋模型（Collobert et al., 2011），即对词向量平均或者对词向量的每一维取最大等方式：

$$\mathbf{ph}_i = \frac{1}{i} \sum_{k=1}^{i} \boldsymbol{x}(w_k) \tag{3.53}$$

$$\mathbf{ph}_i = \max_{k=1}^{d} (\boldsymbol{x}(w_1)_k, \boldsymbol{x}(w_2)_k, \cdots, \boldsymbol{x}(w_i)_k) \tag{3.54}$$

当然，这种方法不考虑短语中不同词的权重，而且没有对词的顺序进行建模。针对前者，可以在对词向量平均的基础上添加词的权重信息：

$$\mathbf{ph}_i = \sum_{k=1}^{i} v_k \boldsymbol{x}(w_k) \tag{3.55}$$

其中，v_k 可以是词 w_k 对应的词频或 TF-IDF 等信息，或者可采用字词混合模型中的门限机制控制不同词对短语表示的贡献。

3.4.2 基于自动编码器的分布式表示

正如前面所述，基于词袋模型的短语表示方法还存在另一个问题，即无法捕捉短语中的词序信息。在很多情形下，词序不同，短语的语义完全不同。例如，两个短语“猫吃鱼”和“鱼吃猫”使用相同的三个词语，语义却完全相反。因此，短语的分布式语义表示学习需要对词语的顺序进行有效建模。本节介绍短语表示学习的一种典型方法，即递归自动编码器（recursive autoencoder, RAE）（Socher et al., 2011b）。

顾名思义，递归自动编码器就是以递归的方式自底向上不断地合并两个子节点的向量表示，直至获得短语的向量表示。图 3.12 给出了一个递归自动编码器应用于二叉树的

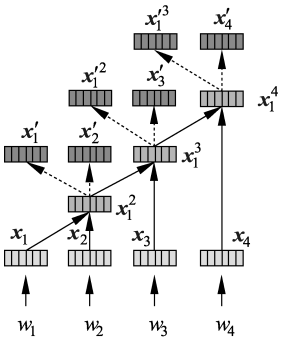


图 3.12 递归自动编码器示意图

例子，其中树上的每个节点都采用相同的标准自动编码器。标准自动编码器的目的是学习给定输入的一个精简、抽象的向量表达。例如，对于图 3.12 中前两个输入词对应的向量 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$，标准自动编码器将利用如下的方式学习一个抽象表示 $\boldsymbol{x}_1^2$：

$$\boldsymbol{x}_1^2 = f\left(\boldsymbol{W}^{(1)}\left[\boldsymbol{x}_1; \boldsymbol{x}_2\right] + \boldsymbol{b}^{(1)}\right) \tag{3.56}$$

其中，$\boldsymbol{W}^{(1)} \in \mathbb{R}^{d \times 2d}$，$\boldsymbol{b}^{(1)} \in \mathbb{R}^d$，$f(\cdot) = \tanh(\cdot)$，即输入两个 d 维向量 $\boldsymbol{x}_1$ 和 $\boldsymbol{x}_2$，输出一个 d 维向量 $\boldsymbol{x}_1^2$，并且要求 $\boldsymbol{x}_1^2$ 是 $\boldsymbol{x}_1$ 与 $\boldsymbol{x}_2$ 的一个压缩抽象表示。为了验证 $\boldsymbol{x}_1^2$ 的质量，可以从输出 $\boldsymbol{x}_1^2$ 重构出输入：

$$\left[\boldsymbol{x}_1'; \boldsymbol{x}_2'\right] = f\left(\boldsymbol{W}^{(2)}\boldsymbol{x}_1^2 + \boldsymbol{b}^{(2)}\right) \tag{3.57}$$

其中，$\boldsymbol{W}^{(2)} \in \mathbb{R}^{2d \times d}$，$\boldsymbol{b}^{(1)} \in \mathbb{R}^{2d}$，$f(\cdot) = \tanh(\cdot)$。标准自动编码器要求输入 $[\boldsymbol{x}_1; \boldsymbol{x}_2]$ 和重构输入 $[\boldsymbol{x}_1'; \boldsymbol{x}_2']$ 之间的误差越小越好：

$$E_{\text{rec}}\left(\left[\boldsymbol{x}_1; \boldsymbol{x}_2\right]\right) = \frac{1}{2}\| \left[\boldsymbol{x}_1; \boldsymbol{x}_2\right] - \left[\boldsymbol{x}_1'; \boldsymbol{x}_2'\right] \|^2 \tag{3.58}$$

将 $\boldsymbol{x}_1^2$ 和 $\boldsymbol{x}_3$ 作为输入，相同的自动编码器可以获得短语 $\boldsymbol{w}_1^3$ 的表示 $\boldsymbol{x}_1^3$。然后以 $\boldsymbol{x}_1^3$ 和 $\boldsymbol{x}_4$ 作为输入，可以得到整个短语的表示 $\boldsymbol{x}_1^4$。

作为一种无监督方法，递归自动编码器以最小化短语的重构误差之和作为目标函数：

$$E_\theta\left(\text{ph}_i\right) = \underset{\text{bt} \in A(\text{ph}_i)}{\text{argmin}} \sum_{\text{nd} \in \text{bt}} E_{\text{rec}}\left(\text{nd}\right) \tag{3.59}$$

其中，$A\left(\text{ph}_i\right)$ 表示短语 ph_i 对应的所有可能的二叉树，nd 表示特定二叉树 bt 上的任意一个节点，$E_{\text{rec}}\left(\text{nd}\right)$ 表示节点 nd 的重构误差。

为了检验整个短语的语义向量表示的质量，可以测试语义相近的短语在语义向量空间中能否聚集在一起。假设用于训练的短语集合为 $S\left(\text{ph}\right)$，对于一个未见的短语 ph^*，利用短语向量之间的余弦距离度量任意两个短语之间的语义相似度，从 $S\left(\text{ph}\right)$ 中搜索与 ph^* 相似的短语列表 $\text{List}\left(\text{ph}^*\right)$，检验 $\text{List}\left(\text{ph}^*\right)$ 与 ph^* 是否真正的语义相近。表 3.2 的第一列给出了 4 个不同长度的英文测试短语，第二列展示了无监督递归自动编码器 RAE 能够找到的向量空间中相近的候选短语列表。

表 3.2 RAE 和 BRAE 在短语语义表示方面的对比

新输入短语	RAE	BRAE
military force	core force main force labor force	military power military strength armed forces
at a meeting	to a meeting at a rate a meeting	at the meeting during the meeting at the conference
do not agree	one can accept i can understand do not want	do not favor will not compromise not to approve
each people in this nation	each country regards each country has its each other, and	every citizen in this country all the people in the country people all over the country

可以发现，RAE 能够在一定程度上捕捉短语的结构信息，例如，“military force”和“labor force”以及“do not agree”和“do not want”等。但是，RAE 在编码短语的语义信息方面比较欠缺。

当然，如果存在一些短语，有正确的语义向量表示作为监督信息，就可以采用有监督的递归自动编码器学习短语的语义表示模型。但是，正确的语义表示在现实中并不存在。为了让短语的向量表示刻画足够的语义信息，Zhang 等提出了一种双语约束的递归自动编码器框架（Zhang et al., 2014），其基本假设是：两个互为翻译的短语具有相同的语义，那么它们应该共享相同的向量表示。基于这个前提假设，可以采用协同训练（co-training）的思想同时学习两种语言的短语向量表示。首先，利用两个递归自动编码器以无监督方式学习语言 X 和语言 Y 中短语的初始表示，然后，以最小化语言 X 和语言 Y 中互译短语 $\left(\mathrm{ph}_x, \mathrm{ph}_y\right)$ 之间的语义距离为目标函数，优化两种语言的递归自动编码器网络。图 3.13 展示了该方法的基本框架，其中 $f(\boldsymbol{x}_1^3, \boldsymbol{y}_1^4)$ 表示源语言短语 $\boldsymbol{x}_1^3$ 和目标语言短语 $\boldsymbol{y}_1^4$ 之间的语义距离，θ 表示神经网络参数。

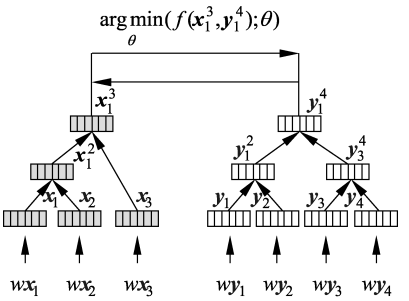


图 3.13 双语约束的递归自动编码器示意图

该方法的目标函数包括两部分：一部分是递归自动编码器的重构误差，另一部分是互译短语之间的语义误差：

$$E\left(\mathrm{ph}_x, \mathrm{ph}_y; \theta\right) = \alpha E_{\mathrm{REC}}\left(\mathrm{ph}_x, \mathrm{ph}_y; \theta\right) + (1-\alpha) E_{\mathrm{SEM}}\left(\mathrm{ph}_x, \mathrm{ph}_y; \theta\right) \tag{3.60}$$

其中，$E_{\text{REC}}\left(\text{ph}_x, \text{ph}_y; \theta\right)$ 表示两个短语的重构误差，$E_{\text{SEM}}\left(\text{ph}_x, \text{ph}_y; \theta\right)$ 表示互译短语之间的语义误差，α 调节重构误差和语义误差之间的权重。$E_{\text{REC}}\left(\text{ph}_x, \text{ph}_y; \theta\right)$ 包括两个短语的重构误差：

$$E_{\text{REC}}\left(\text{ph}_x, \text{ph}_y; \theta\right) = E_{\text{REC}}\left(\text{ph}_x; \theta\right) + E_{\text{REC}}\left(\text{ph}_y; \theta\right) \tag{3.61}$$

每个短语重构误差的计算方式与无监督递归自动编码器的计算方法相同。$E_{\text{SEM}}(\text{ph}_x, \text{ph}_y; \theta)$ 包含两个方向的语义误差：

$$E_{\text{SEM}}\left(\text{ph}_x, \text{ph}_y; \theta\right) = E_{\text{SEM}}\left(\text{ph}_x|\text{ph}_y; \theta\right) + E_{\text{SEM}}\left(\text{ph}_y|\text{ph}_x; \theta\right) \tag{3.62}$$

$$E_{\text{SEM}}\left(\text{ph}_x|\text{ph}_y; \theta\right) = \frac{1}{2}\|\boldsymbol{x}\left(\text{ph}_x\right) - f\left(\boldsymbol{W}_x^l \boldsymbol{y}\left(\text{ph}_y\right) + \boldsymbol{b}_x^l\right)\|^2 \tag{3.63}$$

$$E_{\text{SEM}}\left(\text{ph}_y|\text{ph}_x; \theta\right) = \frac{1}{2}\|\boldsymbol{y}\left(\text{ph}_y\right) - f\left(\boldsymbol{W}_y^l \boldsymbol{x}\left(\text{ph}_x\right) + \boldsymbol{b}_y^l\right)\|^2 \tag{3.64}$$

对于包括 N 个互译短语的短语集合 $\left(\text{ph}_x, \text{ph}_y\right)$，希望在整个数据集上的误差最小：

$$J_{\text{BRAE}}\left(\text{PH}_x, \text{PH}_y; \theta\right) = \frac{1}{N} \sum_{\left(\text{ph}_x, \text{ph}_y\right) \in (\text{PH}_x, \text{PH}_y)} E\left(\text{ph}_x, \text{ph}_y; \theta\right) + \frac{\lambda}{2}\|\theta\|^2 \tag{3.65}$$

其中，第二项表示参数的正则化项（regularization term）。当然，在最小化互译短语的语义距离的同时，也可以最大化非互译短语的语义距离：

$$E_{\text{SEM}}^*\left(\text{ph}_x|\text{ph}_y; \theta\right) = \max\left\{0, E_{\text{SEM}}\left(\text{ph}_x|\text{ph}_y; \theta\right) - E_{\text{SEM}}\left(\text{ph}_x|\text{ph}_y'; \theta\right) + 1\right\} \tag{3.66}$$

其中，$\left(\text{ph}_x, \text{ph}_y\right)$ 是互译短语，$\left(\text{ph}_x, \text{ph}_y'\right)$ 为随机采样的非互译短语。通过协同训练机制，最终可得到两种语言的短语表示模型。

表 3.2 中第三列展示了 BRAE 模型的效果。与无监督的 RAE 相比，BRAE 能够编码短语的语义信息。例如，输入短语“do not agree”，BRAE 能够为其找到语义相近但用词差别较大的短语：“will not compromise”和“not to approve”。可见，双语约束的递归自动编码器 BRAE 能够学习较为准确的短语语义向量表示。

3.5 句子的分布式表示

由于词和短语往往不是文本挖掘任务处理的直接对象，因此，对于词和短语的表示学习主要还是采用通用（或任务无关）的分布式表示方法。相对而言，句子是很多文本挖掘任务的直接处理对象，例如，面向句子的文本分类、情感分析和蕴涵推断等。所以，句子的分布式表示学习至关重要。通常有两大类句子表示方法，一类是通用的，另一类则是任务相关的。

3.5.1 通用的句子表示

通用的句子表示几乎都是以无监督方法为核心思想，设计简单的基于神经网络的句子表示模型，在大规模句子集合 $D = \{w_{i1}^{m_i}\}_{i=1}^{M}$ 的训练数据上优化神经网络参数。以下介绍三种典型的通用句子表示方法。

1. PV-DM 和 PV-DBOW 模型

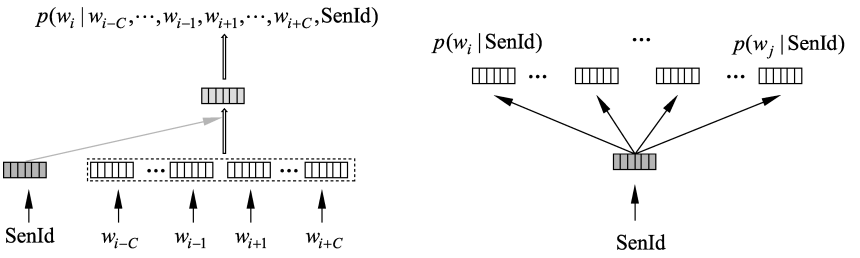
Le 和 Mikolov 于 2014 年对词表示学习中的 CBOW 模型和 Skip-gram 模型进行了扩展，使其可以同时学习词向量和句子向量（Le and Mikolov, 2014）。对于集合 D 中的 M 个句子，按照顺序，每个句子 D_i 对应一个序号 i，该序号 i 可以唯一代表该句子。假设我们希望句子向量的维度为 p，那么训练集中所有句子的向量对应一个矩阵 $\mathbf{PV} \in \mathbb{R}^{M\times p}$。序号为 i 的句子对应的向量是 $\mathbf{PV}$ 中的第 i 行。

对 CBOW 词表示模型的扩展，形成了句子表示模型 PV-DM（paragraph vector with sentence as distributed memory）。如图 3.14（a）所示，PV-DM 模型将上下文所在的句子视为一个记忆单元，捕捉当前上下文缺失的信息。对于任意一个 n 元组 $(w_i, C) = w_{i-C}\cdots w_{i-1}w_i w_{i+1}\cdots w_{i+C}$ 和该 n 元组所在的句子序号 SenId，将 SenId 和 $\mathrm{WC} = w_{i-C}\cdots w_{i-1}w_{i+1}\cdots w_{i+C}$ 作为输入，计算句子和上下文词的平均词向量（或采用向量拼接的方式）：

$$\boldsymbol{h} = \frac{1}{2C+1}\left(\boldsymbol{e}\left(\mathrm{SenId}\right) + \sum_{i-C\leqslant k\leqslant i+C, k\neq i} \boldsymbol{e}\left(w_k\right)\right) \tag{3.67}$$

其中，$\boldsymbol{e}\left(\mathrm{SenId}\right)$ 表示 $\mathbf{PV}$ 中 SenId 对应的句向量。中心词的概率 $p\left(w_i | w_{i-C},\cdots,w_{i-1},w_{i+1},\cdots,w_{i+C},\mathrm{SenId}\right)$ 计算方法、目标函数和训练过程均与 CBOW 模型一致。

对 Skip-gram 模型扩展之后，形成了句子表示模型 PV-DBOW（distributed bag-of-words version of paragraph vector）。如图 3.14（b）所示，该模型以句子为输入，以句子中随机抽样的词为输出，即要求句子能够预测句中的任意词。其目标函数设计和训练方式与 Skip-gram 模型相同。



(a) PV-DM句子表示模型　　(b) PV-DBOW句子表示模型

图 3.14　PV-DM 模型和 PV-DBOW 模型

PV-DM 和 PV-DBOW 两个模型简单有效，但是仅能够对训练数据中的句子学习对应的向量表示。如果希望获得未见测试句子的向量表示，则需要将该句子放入训练集中重新训练模型。所以，这类模型的泛化性能受到了一定的限制。

2. 基于词袋模型的分布式表示

基于语义组合的通用句子表示方法是目前研究的一个热点。其中一类方法是基于词袋模型进行句子表示，该方法认为句子的语义是句中词汇语义的简单组合，最简单就是

采用词向量平均的方法：

$$\boldsymbol{e}_s = \frac{1}{n}\sum_{k=1}^{n}\boldsymbol{e}(w_k) \tag{3.68}$$

其中，$\boldsymbol{e}(w_k)$ 表示词 w_k 对应的词向量，可通过词向量学习方法获得；n 表示句子的长度；$\boldsymbol{e}_s$ 是句子的向量表示。由于句子中不同词对句子语义的贡献也不尽相同，例如，句子“明天将在北京举行‘一带一路’论坛”中，“一带一路”显然对句子语义的贡献更大。因此，在简单组合词汇语义时，如何为每个词赋予合适的权重是这类方法研究的重点，即

$$\boldsymbol{e}_s = \sum_{k=1}^{n} v_k\boldsymbol{e}(w_k) \tag{3.69}$$

其中，v_k 表示词 w_k 的权重。可以采用 TF-IDF 值或信息论中的自信息（self-information）等统计量近似 v_k。（Wang et al., 2017b）提出了一种基于自信息的权重计算方法，通过如下方式计算 v_k：

$$v_k = \frac{\exp(\mathrm{si}_k)}{\sum_{i=1}^{n}\exp(\mathrm{si}_i)} \tag{3.70}$$

其中，$\mathrm{si}_k = -\log\left(P\left(w_k|w_1w_2\cdots w_{k-1}\right)\right)$，表示词 w_k 的自信息，可通过语言模型进行估计。词 w_k 的自信息越大，表明该词所携带的信息量越多，所以在句子表示中应该被赋予更大的权重。这类基于词袋模型的句子表示方法虽然思想简单，但是在相似句子判别、文本蕴涵等自然语言处理任务中表现出很强的竞争力。

3. Skip-Thought 模型

Skip-Thought 方法是另一类基于语义组合的句子表示方法（Kiros et al., 2015），该方法类似于 PV-DBOW 模型（distributed bag-of-words version of paragraph vector），其基本思想也来源于 Skip-gram 模型，但不同于 PV-DBOW 模型利用句子预测句中的词，Skip-Thought 模型（Skip-Thought model）利用当前句子 D_k 预测前一个句子 D_{k-1} 和后一个句子 D_{k+1}。该模型认为，文本中连续出现的句子 $D_{k-1}D_kD_{k+1}$ 表达的意思比较接近，因此，根据句子 D_k 的语义可以重构出前后两个句子。

图 3.15 给出了 Skip-Thought 模型的示意图。该模型有两个核心模块：一个负责对当前句子 D_k 进行编码，另一个负责从 D_k 的语义表示解码生成 D_{k-1} 和 D_{k+1}。编码器采用基于语义组合的循环神经网络，每个神经单元采用门控循环单元（GRU）。编码过程与循环神经网络语言模型一致，这里不再赘述。如图 3.15 左侧所示，得到句中每个位置的隐藏表示 $\boldsymbol{h}_i^k$ 后，最后一个位置对应的隐藏表示 $\boldsymbol{h}_n^k$ 将作为整个句子的语义编码表示。

解码器类似于基于 GRU 的神经网络语言模型，唯一的区别在于每个时刻的输入除了上一时刻的隐藏表示 $\boldsymbol{h}_{j-1}$ 和输出 w_{j-1} 之外，还有句子 D_k 的隐藏表示 $\boldsymbol{h}_n^k$，每个时间节点 GRU 单元的计算过程如下（以预测前一个句子为例）：

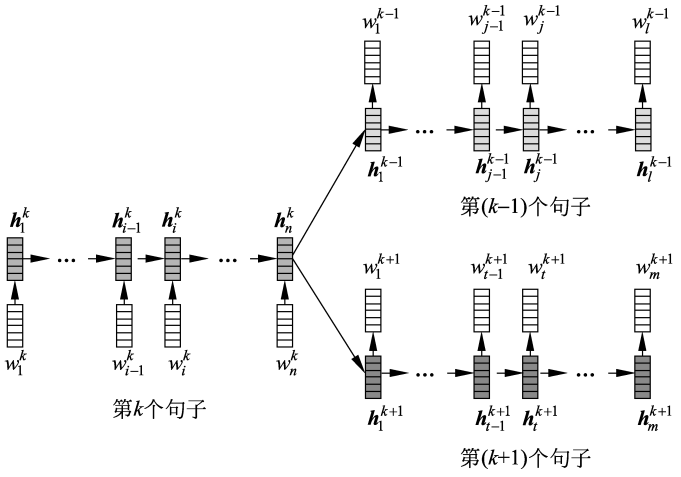


图 3.15 Skip-Thought 句子表示模型

$$\boldsymbol{r}^j = \sigma\left(\boldsymbol{W}_r^{k-1}\boldsymbol{e}\left(w_{j-1}^{k-1}\right) + \boldsymbol{U}_r^{k-1}\boldsymbol{h}_{j-1}^{k-1} + \boldsymbol{C}_r^{k-1}\boldsymbol{h}_n^k + \boldsymbol{b}_r^{k-1}\right) \tag{3.71}$$

$$\boldsymbol{z}^j = \sigma\left(\boldsymbol{W}_z^{k-1}\boldsymbol{e}\left(w_{j-1}^{k-1}\right) + \boldsymbol{U}_z^{k-1}\boldsymbol{h}_{j-1}^{k-1} + \boldsymbol{C}_z^{k-1}\boldsymbol{h}_n^k + \boldsymbol{b}_z^{k-1}\right) \tag{3.72}$$

$$\tilde{\boldsymbol{h}}_j = \tanh\left(\boldsymbol{W}\boldsymbol{e}\left(w_{j-1}^{k-1}\right) + \boldsymbol{U}\left(\boldsymbol{r}^j \odot \boldsymbol{h}_{j-1}^{k-1}\right) + \boldsymbol{C}^{k-1}\boldsymbol{h}_n^k + \boldsymbol{b}\right) \tag{3.73}$$

$$\boldsymbol{h}_j^{k-1} = \boldsymbol{z}^j \odot \tilde{\boldsymbol{h}}_j + \left(1 - \boldsymbol{z}^j\right) \odot \boldsymbol{h}_{j-1}^{k-1} \tag{3.74}$$

给定 $\boldsymbol{h}_j^{k-1}$、已经产生的词语序列 $w_1^{k-1}w_2^{k-1}\cdots w_{j-1}^{k-1}$ 和句子 D_k 的隐藏表示 $\boldsymbol{h}_n^k$，生成下一个词语 w_j^{k-1} 的概率为

$$p\left(w_j^{k-1}|w_{<j}^{k-1}, \boldsymbol{h}_n^k\right) \propto \exp\left(\boldsymbol{e}\left(w_j^{k-1}\right), \boldsymbol{h}_j^{k-1}\right) \tag{3.75}$$

后一个句子 D_{k+1} 的计算过程类似。

Skip-Thought 模型训练的目标函数为

$$\sum_{k=1}^{M}\left\{\sum_{j=1}^{l} p\left(w_j^{k-1}|w_{<j}^{k-1}, \boldsymbol{h}_n^k\right) + \sum_{t=1}^{m} p\left(w_t^{k+1}|w_{<t}^{k+1}, \boldsymbol{h}_n^k\right)\right\} \tag{3.76}$$

其中，M 为训练集合中句子的数目，l 和 m 分别是前一个句子和后一个句子的长度。

Skip-Thought 模型充分结合了语义组合思想和分布式假说。如果训练语料都是由连续文本形式构成的，那么 Skip-Thought 模型可以获得高质量的句子向量表示。

3.5.2 任务相关的句子表示

任务相关的句子表示以具体任务的性能指标为优化目标，例如，在句子级的情感分析任务中，句子的向量表示最终是为了预测该句子的情感极性。以下分别以递归神经网络（recursive neural network, RNN）（Socher et al., 2013）和卷积神经网络（convolutional neural networks, CNN）（Kim, 2014）为例介绍两种任务相关的句子表示学习方法。

1. 基于递归神经网络的句子表示方法

递归神经网络是一种适合于树结构的深度学习模型。给定子节点的向量表示，递归神经网络自底向上地递归学习父节点的向量表示，直至当前父节点覆盖整个句子。图 3.16 分别给出了同一个句子对应的循环神经网络表示（图 3.16（a））和递归神经网络表示（图 3.16（b））。给定一个句子，可以首先通过句法分析技术获得该句子的树结

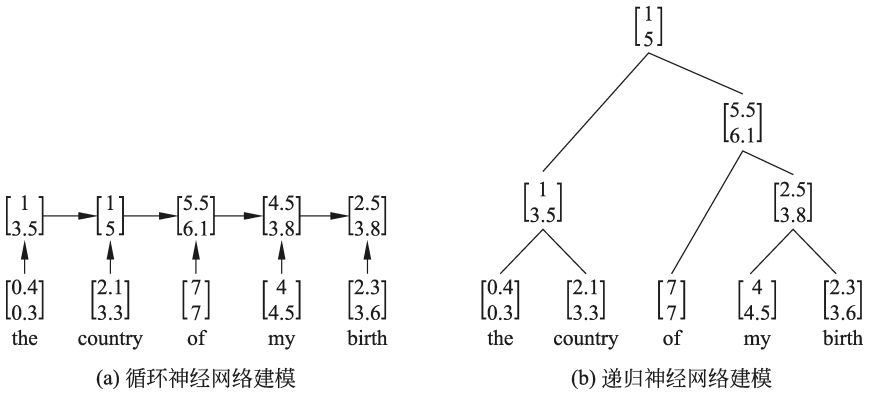


图 3.16 针对句子的两种神经网络建模方法示例

构，通常是一棵二叉树。以图 3.17 所示的句子及其对应的二叉树为例，叶子节点对应每个词的 d 维词向量[①]，递归神经网络沿着树结构合并叶子节点的词向量，分别得到词组 w_1^2，w_3^4 和 w_5^6 的向量表示 $\boldsymbol{x}_1^2$，$\boldsymbol{x}_3^4$ 和 $\boldsymbol{x}_5^6$：

$$\boldsymbol{x}_1^2 = f\left(\boldsymbol{W}_1^{(1)}\left[\boldsymbol{x}_1; \boldsymbol{x}_2\right] + \boldsymbol{b}_1^{(1)}\right) \tag{3.77}$$

$$\boldsymbol{x}_3^4 = f\left(\boldsymbol{W}_2^{(1)}\left[\boldsymbol{x}_3; \boldsymbol{x}_4\right] + \boldsymbol{b}_2^{(1)}\right) \tag{3.78}$$

$$\boldsymbol{x}_5^6 = f\left(\boldsymbol{W}_3^{(1)}\left[\boldsymbol{x}_5; \boldsymbol{x}_6\right] + \boldsymbol{b}_3^{(1)}\right) \tag{3.79}$$

然后，以子节点 $\boldsymbol{x}_3^4$ 和 $\boldsymbol{x}_5^6$ 为输入，获得词组 w_3^6 对应的向量表示 $\boldsymbol{x}_3^6$：

$$\boldsymbol{x}_3^6 = f\left(\boldsymbol{W}_1^{(2)}\left[\boldsymbol{x}_3^4; \boldsymbol{x}_5^6\right] + \boldsymbol{b}_1^{(2)}\right) \tag{3.80}$$

最后，以子节点 $\boldsymbol{x}_1^2$ 和 $\boldsymbol{x}_3^6$ 为输入，递归神经网络得到整个句子的向量表示 $\boldsymbol{x}_1^6$：

$$\boldsymbol{x}_1^6 = f\left(\boldsymbol{W}_1^{(3)}\left[\boldsymbol{x}_1^2; \boldsymbol{x}_3^6\right] + \boldsymbol{b}_1^{(3)}\right) \tag{3.81}$$

上述公式中的参数矩阵 $\boldsymbol{W}_1^{(1)}, \boldsymbol{W}_2^{(1)}, \boldsymbol{W}_3^{(1)}, \boldsymbol{W}_1^{(2)}, \boldsymbol{W}_1^{(3)} \in \mathbb{R}^{d\times 2d}$，偏置 $\boldsymbol{b}_1^{(1)}, \boldsymbol{b}_2^{(1)}, \boldsymbol{b}_3^{(1)}, \boldsymbol{b}_1^{(2)}, \boldsymbol{b}_1^{(3)} \in \mathbb{R}^d$。如果是预测句子的情感极性（正面、负面或中性），$\boldsymbol{x}_1^6$ 将作为句子的抽象特征表示通过 softmax 函数计算情感极性的概率分布：

$$\boldsymbol{t} = \text{softmax}\left(\boldsymbol{W}\boldsymbol{x}_1^6 + \boldsymbol{b}\right) \tag{3.82}$$

① 在任务相关的句子表示学习中，可以随机初始化词向量，也可以采用 Skip-gram 等模型预训练获得的词向量作为初始值，然后在句子表示学习的过程中进一步优化底层词向量。

其中，$\boldsymbol{W} \in \mathbb{R}^{3\times d}$，$\boldsymbol{b} \in \mathbb{R}^3$，数字 3 对应情感极性的维度（1 表示正面，$-1$ 表示负面，0 表示中性）。给定 n 组“句子，情感极性”的训练数据 $D = (D_i, L_i)_{i=1}^n$，递归神经网络以最小化交叉熵为目标函数优化网络参数 θ（包括参数矩阵、偏置和词向量）：

$$\theta^* = \underset{\theta}{\operatorname{argmin}} - \sum_{i=1}^{n} \sum_{l} \delta_{L_i}(l) \log p(D_i, l) \tag{3.83}$$

其中，$L_i \in \{-1, 0, 1\}$，如果 $l = L_i$，$\delta_{L_i}(l) = 1$，否则 $\delta_{L_i}(l) = 0$；$p(D_i, l)$ 表示 $\boldsymbol{t}$ 中情感极性 l 对应的概率。

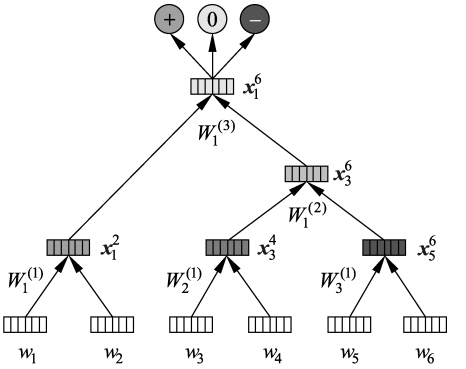


图 3.17 基于递归神经网络的句子表示方法

从图 3.17 可以发现，递归神经网络与递归自动编码器非常相似，主要区别有三点：第一，递归神经网络以具体的一棵二叉树为输入，而递归自动编码器需要搜索一棵最佳的二叉树；第二，递归神经网络不需要在每个节点计算重构误差；第三，递归神经网络在不同的节点可使用相同的参数，也可以根据子节点类型采用不同的参数，例如，参数矩阵 $\boldsymbol{W}_1^{(1)}, \boldsymbol{W}_2^{(1)}, \boldsymbol{W}_3^{(1)}, \boldsymbol{W}_1^{(2)}, \boldsymbol{W}_1^{(3)}$ 和偏置 $\boldsymbol{b}_1^{(1)}, \boldsymbol{b}_2^{(1)}, \boldsymbol{b}_3^{(1)}, \boldsymbol{b}_1^{(2)}, \boldsymbol{b}_1^{(3)}$ 可以相同，也可以不同。

2. 基于卷积神经网络的句子表示方法

递归神经网络基于树结构，适合于对词序和层次化结构有依赖的任务，例如，情感分析和句法分析等。对于句子的主题分类任务，句子中的某些关键信息对于主题类别预测起着决定性的作用，因此，卷积神经网络成为解决这类任务的经典模型。如图 3.18 所示，对于一个句子，卷积神经网络以每个词的词向量作为输入，通过顺序地对上下文窗口[①]进行卷积（convolution）总结局部信息，并利用池化层（pooling）提取全局的重要信息，再经过其他网络层（卷积池化层、Dropout 层和线性层等），得到固定维度的句子向量表示，以刻画句子全局性的语义信息。

形式化地，给定包含 n 个词的句子 $w_1 w_2 \cdots w_n$，每个词首先利用预训练（pre-train）或随机初始化的词向量矩阵 $\boldsymbol{L} \in \mathbb{R}^{|V|\times d}$ 映射为词向量列表 $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n]$。对于

① 在窗口大小 h 的选择方面，可采用多组窗口（例如，图 3.18 中 $h = 3, 5$）进行全局的信息提取。

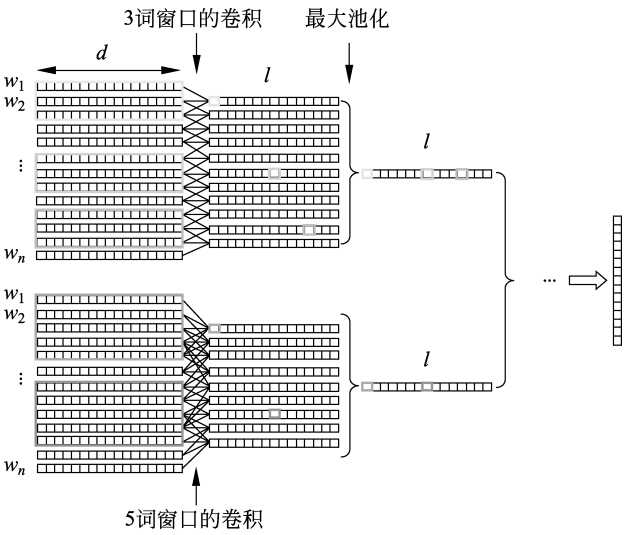


图 3.18 基于卷积神经网络的句子表示方法

任意一个 h 长度的窗口 $\boldsymbol{x}_{i:i+h-1}$，卷积层采用卷积算子（convolution operator）[①] F_t（$1 \leqslant t \leqslant T$，$T$ 表示卷积算子数目）得到一个局部特征 y_i^t：

$$y_i^t = F_t\left(\boldsymbol{W}\boldsymbol{x}_{i:i+h-1} + b\right) \tag{3.84}$$

其中，$F_t(\cdot)$ 表示非线性激活函数，$\boldsymbol{W} \in \mathbb{R}^{hd}$，$b \in \mathbb{R}$，$y_i^t \in \mathbb{R}$。卷积算子 F_t 从 $\boldsymbol{x}_{1:h-1}$ 到 $\boldsymbol{x}_{n-h+1:n}$ 遍历整个句子，得到特征列表 $\boldsymbol{y}^t = \left[y_1^t, y_2^t, \cdots, y_{n-h+1}^t\right]$。可见，$\boldsymbol{y}^t \in \mathbb{R}^{n-h+1}$ 是一个不定长的向量，维度直接取决于句子长度 n，句子长度从几个词到上百个词，$\boldsymbol{y}^t$ 的维度也将随着句长改变而动态变化。

为了将不定长的 $\boldsymbol{y}^t$ 转换为定长的输出，池化成为不可或缺的操作。最大池化是最流行的池化方法（Collobert et al., 2011; Kim, 2014），该方法认为 $\hat{y}^t = \max(\boldsymbol{y}^t)$ 代表了卷积算子 F_t 在整个句子上获得的最重要特征。T 个卷积算子将得到一个 T 维的特征向量 $\boldsymbol{y} = \left[\hat{y}^1, \hat{y}^2, \cdots, \hat{y}^T\right]$。

窗口大小 h 是一个经验值，为使模型具有一定的鲁棒性，卷积神经网络一般尝试多个不同尺度的窗口 h，例如，图 3.18 分别采用了 $h=3$ 和 $h=5$，其中每个窗口对应一个 T 维的特征向量 $\boldsymbol{y} = \left[\hat{y}^1, \hat{y}^2, \cdots, \hat{y}^T\right]$。之后经过其他网络层便可以获得句子定长的向量表示。如果应用于句子主题分类任务，则可以在训练数据上采用类似于情感分析任务中的最小化交叉熵的目标函数优化卷积神经网络中的所有参数。

3.6 文档的分布式表示

在文本分类、情感分析、文本摘要和篇章分析等诸多自然语言处理任务中，文档是最常见的直接处理对象。对文档的深度理解是实现这些任务的关键，而文档理解的前提

① 一般也称为过滤器，完成对上下文窗口中信息的过滤。

是对文档进行表示。由于文档的分布式表示可以捕捉更多全局的语义信息，因此成为一个重要的研究方向。如何从词、短语和句子的分布式表示学习文档的分布式表示是整个问题的关键。本节将从通用模型和面向任务的模型两个角度介绍文档的分布式表示方法。

3.6.1 通用的文档分布式表示

1. 基于词袋的文档表示

在通用的文档分布式表示中，文档可视为一个特殊的句子，即所有句子的自然拼接。因此，可以采用类似于句子的分布式表示方法学习文档的分布式表示。例如，基于组合语义的词袋模型可以快速地从词的分布式表示获得文档 $D=(D_i)_{i=1}^{M}$ 的分布式表示：

$$\boldsymbol{e}_D=\sum_{k=1}^{|D|}v_k\boldsymbol{e}\left(w_k\right) \tag{3.85}$$

其中，v_k 表示词 w_k 的权重，$|D|$ 表示文档 D 中不同词的数目。可以采用平均词向量方法 $v_k=\dfrac{1}{|D|}$，或者采用加权的词向量方法 $v_k=\text{TF-IDF}\left(w_k\right)$。这类方法简单高效，但不足之处在于既没有考虑句子内部词语之间的顺序，也没有考虑文档内部句子和句子之间的相互关系。

2. 基于层次化自编码器的文档表示

针对词袋模型表示能力不足的问题，Li 等提出了一种层次化的自编码器模型（Li et al., 2015），其基本思想是：对 M 个句子的文档 $D=(D_i)_{i=1}^{M}$ 进行编码获得对应的向量表示 $\boldsymbol{e}_D$，若能够从 $\boldsymbol{e}_D$ 重构出文档 D，那么 $\boldsymbol{e}_D$ 就应该是文档 D 正确的分布式表示。

该层次化自编码器模型分为两个部分，一个是从文档 D 到向量表示 $\boldsymbol{e}_D$ 的编码模型，另一个是从向量表示 $\boldsymbol{e}_D$ 到文档 D 的重构模型。在编码模型中，长短时记忆网络（LSTM）首先用于获得每个句子的表示 $\boldsymbol{e}_{s_i}$，然后以句子表示为输入，采用第二层 LSTM 网络对文档中的句子序列进行建模，从而获得文档的表示 $\boldsymbol{e}_D$。其中 $\boldsymbol{e}_{s_i}$ 和 $\boldsymbol{e}_D$ 分别是句子和文档结尾符对应的 LSTM 隐层表示：

$$\boldsymbol{e}_{s_i}=\boldsymbol{h}^{s}_{\text{end}_s}\left(\text{enc}\right) \tag{3.86}$$

$$\boldsymbol{h}^{s}_{t}\left(\text{enc}\right)=\text{LSTM}\left(\boldsymbol{e}_{w_t},\boldsymbol{h}^{s}_{t-1}\left(\text{enc}\right)\right) \tag{3.87}$$

$$\boldsymbol{e}_D=\boldsymbol{h}^{D}_{\text{end}_D}\left(\text{enc}\right) \tag{3.88}$$

$$\boldsymbol{h}^{D}_{t}\left(\text{enc}\right)=\text{LSTM}\left(\boldsymbol{e}_{s_t},\boldsymbol{h}^{D}_{t-1}\left(\text{enc}\right)\right) \tag{3.89}$$

重构网络的目标在于从文档的分布式表示 $\boldsymbol{e}_D$ 重构出文档 D，所采用的方法同样是层次化的 LSTM 模型：首先重构出句子级隐层表示 $\boldsymbol{h}^{s}_{t}\left(\text{dec}\right)$，然后重构出句子 s_t 中的所有词语：

$$\boldsymbol{h}_t^D(\text{dec}) = \text{LSTM}\left(\boldsymbol{e}'_{s_{t-1}}, \boldsymbol{h}_{t-1}^D(\text{dec}), \boldsymbol{c}_t^D\right) \tag{3.90}$$

$$\boldsymbol{h}_t^s(\text{dec}) = \text{LSTM}\left(\boldsymbol{e}_{w_{t-1}}, \boldsymbol{h}_{t-1}^s(\text{dec})\right) \tag{3.91}$$

其中，$\boldsymbol{h}_0^D(\text{dec}) = \boldsymbol{e}_D$，$\boldsymbol{e}'_{s_{t-1}}$ 表示前一个句子结尾符对应的隐层表示，$\boldsymbol{c}_t^D$ 表示编码模型的上下文表示，可通过注意力机制计算：

$$\boldsymbol{c}_t^D = \sum_{k=1}^{M} a_k \boldsymbol{h}_k^D(\text{enc}) \tag{3.92}$$

$$a_k = \frac{\exp(v_k)}{\sum_{k'} \exp(v_{k'})} \tag{3.93}$$

$$v_k = \boldsymbol{v}^{\text{T}} f\left(\boldsymbol{W}_1 \boldsymbol{h}_{t-1}^D(\text{dec}) + \boldsymbol{W}_2 \boldsymbol{h}_k^D(\text{enc})\right) \tag{3.94}$$

其中，a_k 表示编码模型中每个句子的权重，$\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{d\times d}$，$\boldsymbol{v} \in \mathbb{R}^{d\times 1}$。$\boldsymbol{h}_0^s(\text{dec}) = \boldsymbol{e}'_{s_0}$，表示重构句子的隐层表示。依据 $\boldsymbol{h}_t^s$，计算词汇 w_t 的概率：

$$p(w_t|\cdot) = \text{softmax}\left(\boldsymbol{e}_{w_t}, \boldsymbol{h}_t^s(\text{dec})\right) \tag{3.95}$$

上述神经网络在训练过程中以最大化原始文档的似然概率为目标函数，即每个时刻重构的输出词与原始文档对应位置的词应该相同。

在图 3.19 中，文档包含两个句子。首先进行第一层 LSTM 编码，分别得到两个句子的表示 $\boldsymbol{e}_{s_1}$ 和 $\boldsymbol{e}_{s_2}$（句子结束符对应的隐层表示）。然后第二层 LSTM 用于对句子序列 $\boldsymbol{e}_{s_1}$ 和 $\boldsymbol{e}_{s_2}$ 进行编码，得到文档表示 $\boldsymbol{e}_D$。以文档表示 $\boldsymbol{e}_D$ 为输入，采用注意力机制计算编码模型句子级表示 $\boldsymbol{e}_{s_1}$ 和 $\boldsymbol{e}_{s_2}$ 的上下文。之后重构每个句子的隐层表示 $\boldsymbol{h}_t^D(\text{dec})$，并逐词生成重构句子。模型训练结束后，层次化的编码网络就可以获得任意文档的分布式表示 $\boldsymbol{e}_D$。

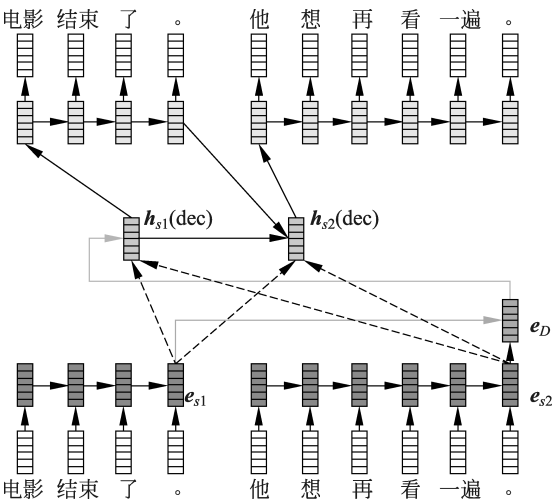


图 3.19 基于自编码器的文档表示方法

3.6.2 任务相关的文档分布式表示

任务相关的文档分布式表示方法以优化任务的性能为最终目标，广泛应用于文本分类和情感分析等任务。本节介绍 Tang 等（2015）提出的一种任务相关的文档表示方法。

在这类方法中，文档被视为句子的有机组合，句子又可以看作词的有机组合。因此，从词到句子和句子到文档的语义组合方式是文档表示方法学习的核心任务。假设文档 $D = (D_i)_{i=1}^{M}$ 由 M 个句子组成，其中第 i 个句子 $D_i = s_i = w_{i,1}, w_{i,2} \cdots w_{i,n}$ 由 n 个词组成。那么，文档的表示学习模型可分为三层：底部的句子表示层、中间的文档表示层和顶部的分类层，如图 3.20 所示。

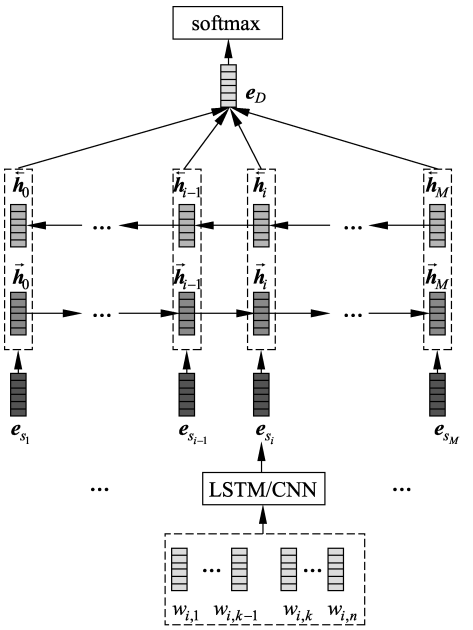


图 3.20 基于层次化模型的文档表示方法

句子表示层学习从词序列 $w_{i,1}w_{i,2}\cdots w_{i,n}$ 到句子 s_i 的语义组合方式。在本章前面几节中已经介绍了循环神经网络、递归神经网络和卷积神经网络等句子表示模型。其中，循环神经网络和卷积神经网络应用得最为广泛。这两种网络都可以用来获得句子的分布式表示：

$$\boldsymbol{e}_{s_i} = \mathrm{LSTM}\left(w_{i,1}w_{i,2}\cdots w_{i,n}\right) \tag{3.96}$$

$$\boldsymbol{e}_{s_i} = \mathrm{CNN}\left(w_{i,1}w_{i,2}\cdots w_{i,n}\right) \tag{3.97}$$

在实际任务中，可以分别尝试上述两种模型，然后选择一种效果更好的方法。

文档表示层用来学习从句子序列 $s_1 s_2 \cdots s_M$ 到文档 D 的语义组合方式，在这一过程中双向循环神经网络是一种常用的方法。以句子的分布式表示 $\boldsymbol{e}_{s_1}\boldsymbol{e}_{s_2}\cdots\boldsymbol{e}_{s_M}$ 为输入，双向 LSTM 模型分别学习每个句子 s_i 的正向隐层表示 $\overrightarrow{\boldsymbol{h}}_i$ 和逆向隐层表示 $\overleftarrow{\boldsymbol{h}}_i$：

$$\overrightarrow{\boldsymbol{h}}_i = \mathrm{LSTM}\left(\boldsymbol{e}_{s_i}, \overrightarrow{\boldsymbol{h}}_{i-1}\right) \tag{3.98}$$

$$\overleftarrow{\boldsymbol{h}}_i = \mathrm{LSTM}\left(\boldsymbol{e}_{s_i}, \overleftarrow{\boldsymbol{h}}_{i+1}\right) \tag{3.99}$$

将双向隐层表示拼接成为句子 s_i 对应的隐层表示 $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$。依据每个句子的隐层表示，可以采用平均策略或注意力机制模型得到文档的表示：

$$\boldsymbol{e}_D = \sum_{i=1}^{M} v_i \boldsymbol{h}_i \tag{3.100}$$

其中，$v_i = \dfrac{1}{M}$，或者 v_i 是由注意力机制模型学习的权重。

给定文档的分布式表示 $\boldsymbol{e}_D$，分类层首先采用一个全连接网络将 $\boldsymbol{e}_D$ 转换为维度为类别数目 C 的分值向量 $\boldsymbol{x} = [x_1, x_2, \cdots, x_C]$，然后 softmax 函数将分值向量 $\boldsymbol{x}$ 转化为类别的概率分布 $\boldsymbol{p} = [p_1, p_2, \cdots, p_C]$：

$$\boldsymbol{x} = f\left(\boldsymbol{W}\boldsymbol{e}_D + \boldsymbol{b}\right) \tag{3.101}$$

$$p_k = \frac{\exp\left(x_k\right)}{\sum_{k'=1}^{C} \exp\left(x_{k'}\right)} \tag{3.102}$$

在文本分类或情感分析等任务中，存在大量的标注数据 $T = \{(D, L)\}$，D 为文档，L 是文档对应的正确类别。训练过程以最小化交叉熵损失为模型优化目标：

$$\mathrm{Loss} = -\sum_{D \in T} \sum_{k=1}^{C} L_k\left(D\right) \log\left(p_k\left(D\right)\right) \tag{3.103}$$

其中，如果文档 D 属于第 k 类，则 $L_k\left(D\right) = 1$；否则，$L_k\left(D\right) = 0$。模型训练后，句子和文档层的网络就可以学习任意文档的分布式表示。

3.7 进一步阅读

由于词是构成短语、句子和篇章的基本语言单元，因此，词的表示学习是基础，也是最受关注的研究方向。词的分布式表示学习的前沿研究主要体现在如下 4 个方面：①如何充分挖掘词的内部结构信息（Xu et al., 2016；Bojanowski et al., 2017；Pinter et al., 2017）；②如何更加有效地利用上下文信息（Ling et al., 2015；Hu et al., 2016；Li et al., 2017a）和词典、知识图谱等外部知识（Wang et al., 2014；Tissier et al., 2017）；③如何更好地解释词向量表示（Arora et al., 2016；Wang et al., 2018）；④如何有效地评价词分布式表示的质量（Yaghoobzadeh and Schutze, 2016）。Lai 等（2016）总结了主流的词表示方法，并提出了如何学习更优词向量的一些设想。

短语、句子和文档的表示学习多集中在语义组合方式的学习上。例如，（Yu and Dredze, 2015）提出了多种特征融合的语义组合函数模型，用来学习短语分布式表示，（Wang and Zong, 2017）对比了不同组合方式在短语表示学习方面的优势和不

足；（Hashimoto and Tsuruoka, 2016）研究了短语是否可由内部词的语义组合而成。句子表示学习更加关注语义组合方式（Gan et al., 2017；Wieting and Gimpel, 2017）和语言学知识的利用问题（Wang et al., 2016d）。文档通常有基于组合语义的分布式表示和基于主题分布的表示两种表示方法，如何将两者优势互补，学习更加准确的文档表示也是一个研究热点（Li et al., 2016b）。

习　　题

3.1　给定以下 4 个文档：

d1 北京理工大学计算机专业创建于 1958 年是中国最早设立计算机专业的教育高校之一

d2 北京理工大学学子在第四届中国计算机博弈锦标赛中夺冠教育

d3 北京理工大学体育馆是 2008 年中国北京奥林匹克运动会的排球预赛场地体育

d4 第五届东亚运动会中国军团奖牌总数创新高男女排球双双夺冠

基于向量空间模型，以文档中出现的所有词作为特征项，分别使用 BOOL 权重、TF 权重、IDF 权重和 IT-IDF 权重，建立 4 个文档的文本表示向量。

3.2　在向量空间模型的特征性权重计算中，TF 和 IDF 的计算都采用了对数形式，请分析说明如果不采用对数形式会遇到哪些问题。

3.3　在向量空间模型中，向量的长度等于词表的规模。如果训练数据中的所有词汇都进行统计，那么词表规模可能会达到十万甚至更多，导致文本表示和计算的效率很低。请阐述有哪些降低词表规模的方法，并说明每种方法的优劣。

3.4　在 C&W 模型中，我们将目标函数设为正样本的得分 $s(w_i, C)$ 大于负样本的得分与特定间隔（例如，1.0）之和 $s(w_i', C) + 1$。请分析为什么需要添加一个特定间隔，如果不添加会导致什么问题。

3.5　在噪声对比估计与负采样方法中，负样本数目 K 是一个需要人工设定的超参数，请分析 K 的大小对效率和效果的影响，对比采用噪声对比估计或负采样方法后计算复杂度的变化情况，并且说明应该如何选择负样本。

3.6　汉语里的很多词语是语义透明的，也就是词语的语义可以由组成该词语的汉字的语义组合获得。请分析在文中介绍的字词混合模型中如何自动地判别一个词语是否语义透明。

3.7　无论是前馈神经网络语言模型还是循环神经网络语言模型，优化目标都是最大化每个时刻标准答案（该时刻应该输出的目标词汇）的概率。请证明最大化对应词汇概率的目标函数等价于最小化交叉熵损失函数。

3.8　在句子的分布式向量表示方法中，卷积神经网络是比较有效的方法之一，但是卷积神经网络中的池化算子主要实现了全局信息汇聚的功能，而忽略了文本中词汇之间的顺序。请设计一种方法，既能够保持卷积神经网络的优势，又能一定程度地建模文本中的语序信息。

第 4 章　预训练语言模型

4.1　概　　述

在第 3 章，我们介绍了单词、短语、句子和文档的分布式表示的各种学习方法。这些分布式表示通常应用于下游文本数据挖掘任务，例如，实体识别、文本情感分析、关系抽取和文本摘要等。虽然这些表示学习方法可以显著改善下游任务的性能，但是下游任务的性能却因一些关键问题受到限制。首先，下游任务采用的神经网络模型层数不能太深，因为下游任务对应的有监督训练数据相对稀缺，导致海量的模型参数在训练中无法得到充分优化。其次，通过学习获得的分布式表示通常是固定的（静态的），不能解决文本的多义性问题。例如，“star”一词可以表示“明星”和“发光的天体”两种含义，而静态的分布式表示不能在动态的上下文中区分不同含义。再次，不同的下游任务通常采用不同的模型进行学习，知识共享没有得到充分利用。

近年来，一种被称作预训练和微调的新范式被提出，并已广泛应用于自然语言处理任务。在执行具体任务之前，研究者们通常设计大规模神经网络模型，并基于海量的无标注文本数据（或称自监督文本数据，可在互联网上轻松获得），按照语言模型或者其他自监督目标函数对神经网络参数进行优化。由于整个训练过程与任何具体任务无关，因此称为预训练。

基于海量文本数据进行预训练而产生的模型将会更加鲁棒，并且模型内部参数能够记忆更多的语言规律。在随后的微调步骤中，可以使用特定任务对应的少量标注数据来微调任务相关的模型参数，从而可以更好地适应特定的下游任务。由于有效地使用了海量无标注数据和与少量任务相关的标注数据，该范式能够在诸多文本处理任务中达到最先进的性能表现。本章简要介绍几种被广泛使用的预训练模型，包括 ELMo，GPT，BERT，XLNet 和 UniLM。

4.2　ELMo：源自语言模型的语境化分布式向量表示

如 4.1 节所述，一个单词可以由其上下文进行表示。因此，单词表示（词向量）的质量至少取决于两个因素。一方面，上下文是否足够丰富，即我们是否有大量的文本数据，其中每个单词都拥有类型丰富的上下文；另一方面，能否很好地捕获和利用上下文。换句话说，如果模型不能有效地利用并表示一个单词的全部上下文，我们就无法得到令人

满意的词向量。当我们把词向量应用于下游任务时，还会出现另一个必须解决的关键问题：词向量是否与下游任务的上下文相关？例如，词向量是基于递归神经网络（RNN）的大型语言模型的副产品。如果直接将预训练的词向量应用于下游任务，那么这种用法只是用到了静态的词向量，与下游任务的上下文无关。但是，如果我们首先使用预训练好的 RNN 语言模型获取测试语句的语境化动态表示，然后将其应用到下游任务中，那么这种用法就是与下游任务上下文相关的。

普遍认为，Peters 等提出的 ELMo[①]模型（Peters et al., 2018）是第一个着眼于解决上述所有问题的模型，它在多个下游文本处理任务中实现了显著的性能提升。ELMo 采用的预训练框架如下：在预训练阶段，利用包含 10 亿词的数据集（包括大约三千万个句子）[②]训练一个基于双向长短时记忆网络（LSTM）的语言模型；在特定的下游任务应用中，首先将待测试的句子输入预训练好的双向 LSTM 中，然后利用神经网络中的动态隐层表示来计算与任务相关和上下文相关的语境化词向量，最后，该语境化词向量在任务相关的模型中进一步微调，以执行特定的文本处理任务。

4.2.1 基于双向 LSTM 的语言模型

ELMo 使用了基于双向 LSTM 的语言模型进行预训练。给定一个句子 (SOS $x_1 \cdots x_{j-1} x_j \cdots x_n$ EOS)（SOS 和 EOS 是表示句子开始和结束的特殊符号），前向的语言模型以左侧的上下文作为历史条件计算 x_j 的概率 $p(x_j \,|\, \text{SOS}\ x_1 \cdots x_{j-1})$，与此同时，后向的语言模型以右侧的上下文作为历史条件计算 x_j 的概率 $p(x_j \,|\, x_{j+1} \cdots x_n\ \text{EOS})$。直观地，左右两个方向的上下文都能够被语言模型捕获。

如图 4.1 所示，底层首先将单词映射至分布式向量表示（ELMo 采用字符级卷积神经网络 CNN 获得词汇表示）。然后，利用 L 层前向和后向的 LSTM 来学习两个方向上的语言模型。为了计算概率 $p(x_j \,|\, \text{SOS}\ x_1 \cdots x_{j-1})$，前向语言模型首先将词向量 $\boldsymbol{x}_{j-1}$ 送入 L 层的前向 LSTM 中，得到顶层表示 $\overrightarrow{\boldsymbol{h}}^L_{j-1}$，然后使用 softmax 函数计算 x_j 的概率。

$$p(x_j \,|\, \text{SOS}\, x_1, \cdots, x_{j-1}) = \text{softmax}(\overrightarrow{\boldsymbol{h}}^L_{j-1}, x_j) = \frac{\overrightarrow{\boldsymbol{h}}^L_{j-1} \cdot \boldsymbol{x}_j}{\sum\limits_{x'} \overrightarrow{\boldsymbol{h}}^L_{j-1} \cdot \boldsymbol{x}'} \tag{4.1}$$

类似地，后向语言模型使用 L 层后向 LSTM 获得顶层表示 $\overleftarrow{\boldsymbol{h}}^L_{j+1}$，并计算概率 $p(x_j \,|\, x_{j+1} \cdots x_n\ \text{EOS})$。双向 LSTM 用 T 个句子（原始的 ELMo 工作中使用了三千万个句子）进行训练，通过最大化前向语言模型和后向语言模型的联合对数似然概率来优化网络参数。

$$\sum_{t=1}^{T} \sum_{j=0}^{n+1} \left(\log p\left(x_j^{(t)} \,|\, \text{SOS}\, x_1^{(t)}, \cdots x_{j-1}^{(t)}; \boldsymbol{\theta} \right) + \log p\left(x_j^{(t)} \,|\, x_{j+1}^{(t)}, \cdots x_n^{(t)}\ \text{EOS}; \boldsymbol{\theta} \right) \right) \tag{4.2}$$

① 全称为 Embeddings from Language Models，代码和模型见 https://allennlp.org/elmo。

② https://github.com/ciprian-chelba/1-billion-word-language-modeling-benchmark。

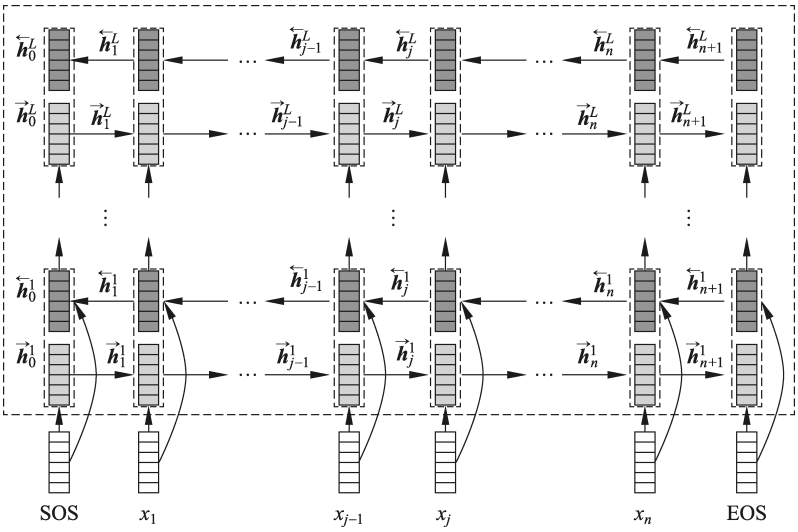


图 4.1 ELMo 模型结构

4.2.2 适应下游任务的语境化 ELMo 词向量

经过预训练，我们得到了双向 LSTM 语言模型以及副产品词向量。如果在下游任务中直接利用词向量 $\boldsymbol{x}_j$，一方面无法区分“star”等多义词的语义，另一方面完全浪费了双向 LSTM 语言模型。ELMo 旨在学习动态词向量，会根据测试句子的上下文得到语境化的词向量。具体地，将下游任务中的每个测试语句输入预训练得到的双向 LSTM 语言模型中，从而得到 $(2L+1)$ 层的表示，包括一个输入层表示和两个 L 层的前向后向 LSTM 隐层表示（在原始的 ELMo 文章中，$L=2$）。$\boldsymbol{x}_j$ 的所有 $(2L+1)$ 层表示可以改写为如下形式：

$$\boldsymbol{R}_j = \{\boldsymbol{x}_j, (\overrightarrow{\boldsymbol{h}}_j^l, \overleftarrow{\boldsymbol{h}}_j^l) \,|\, l=1,2,\cdots,L\} = \{\boldsymbol{h}_j^l \,|\, l=1,2,\cdots,L\} \tag{4.3}$$

其中，$\boldsymbol{h}_j^0 = \boldsymbol{x}_j$ 代表输入层的表示，如果 $l \in \{1,2,\cdots,L\}$，则 $\boldsymbol{h}_j^l = [\overrightarrow{\boldsymbol{h}}_j^l, \overleftarrow{\boldsymbol{h}}_j^l]$。给定下游任务的一个测试句子，双向 LSTM 语言模型将获取 L 层前向和后向的隐层表示；之后通过线性组合得到语境化的 ELMo 词向量表示：

$$\boldsymbol{ELMo}_j^{\text{task}} = \gamma^{\text{task}} \sum_{l=0}^{L} w_l^{\text{task}} \boldsymbol{h}_j^l \tag{4.4}$$

其中，w_l^{task} 决定了每一层隐层表示的贡献，γ^{task} 则表示 ELMo 词向量在特定任务中的重要性。

如图 4.2 所示，图中上半部分表示当采用静态词向量时，多义词“小米”在语义空间中无法区分究竟是公司品牌还是食物种类。ELMo 可以根据具体上下文学习语境化的词向量，从而可以分辨“小米”的具体含义。

在处理下游任务时，语境化 ELMo 词向量通常作为额外特征应用到处理特定文本任务的有监督模型之中。假定在特定下游任务中，测试句子为 $(x_1,\cdots,x_j,\cdots,x_n)$，

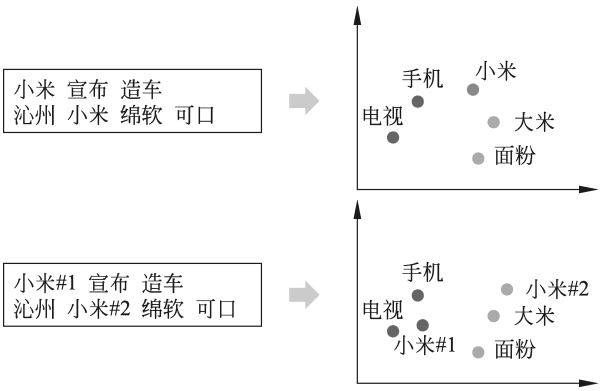


图 4.2 一词多意示意图

基线监督模型（例如，CNN，RNN 或者前馈神经网络）学习到的最终隐层表示为 $(\boldsymbol{h}_1^{\text{task}}, \cdots, \boldsymbol{h}_j^{\text{task}}, \cdots, \boldsymbol{h}_n^{\text{task}})$。可以通过两种方式利用语境化 ELMo 词向量来增强基线监督模型。一方面，ELMo 词向量可以与输入向量 $\boldsymbol{x}_j$ 组合在一起，作为有监督模型新的输入 $[\boldsymbol{x}_j; \boldsymbol{ELMo}_j^{\text{task}}]$。另一方面，ELMo 词向量可以与基线模型得到的最终表示 $\boldsymbol{h}_j^{\text{task}}$ 拼接在一起，即 $[\boldsymbol{h}_j^{\text{task}}; \boldsymbol{ELMo}_j^{\text{task}}]$，这种方式可以直接应用到结果预测之中，而不需要修改基线监督模型的网络结构。

引入语境化 ELMo 词向量来增强模型后，若干文本处理任务取得了显著的性能提升，例如，问答系统、文本蕴涵、语义角色标注、共指消解、命名实体识别和情感分析等。

4.3 GPT：生成式预训练模型

ELMo 预训练模型取得了巨大成功，但是仍然有一些不足之处。首先，ELMo 采用了较浅的两层双向 LSTM，这使得其难以习得海量文本数据中的所有语言规律，因此潜力有限。其次，双向 LSTM 并不是捕获长距离依赖性的最佳方法。例如，当我们需要建模序列中第一个单词和第 n 个单词之间的依赖关系时，无论是前向 LSTM 还是后向 LSTM，都需要 $n-1$ 次迭代才能实现，并且其结果会因为梯度消失的问题受到进一步的影响。再次，双向 LSTM 采用从左往右或者从右往左的链式计算方法，只有当第 $n-1$ 个位置计算完毕后才能计算第 n 个位置，无法采用并行化算法提升模型训练和推断的效率。最后，ELMo 预训练模型的能力尚未得到充分利用。从 4.2 节介绍可知，ELMo 仅用于获取隐层表示，并作为额外的特征应用于下游任务。也就是说，下游监督任务中的微调模型并非 ELMo 采用的双向 LSTM 模型，微调模型的参数需要从头开始学习。

鉴于 EMLo 模型的不足，Radford 等（2018）受到 Transformer（Vaswani et al., 2017）的启发，提出了一种基于注意力机制的深层模型 GPT（generative pre-training，即生成式预训练模型），不仅用于预训练过程，后续的微调过程也采用相同的模型。具体地，GPT 采用了包含 12 个自注意层的 Transformer 解码器，使用前向的语言模型作为

预训练目标，在下游任务中，微调的对象仍然是这 12 层的自注意模型。本节将首先简要介绍 Transformer，然后对 GPT 模型进行概述。

4.3.1 Transformer

Transformer[①]的提出最初是为了解决机器翻译任务，即将源语言句子自动转换成为目标语言句子。Transformer 模型遵循编码器-解码器架构，其中编码器用于学习源语言句子的语义表示，解码器则根据源语言句子的语义表示，从左向右逐词解码生成目标语言句子。

编码器包括 L 层，每层由两个子层组成，分别是自注意力子层和前馈神经网络子层，如图 4.3 中左侧所示。其中，核心的自注意子层通过使用当前层中的第 i 个位置的表示来与包括其自身在内的所有位置的表示计算注意力权重，然后将所得权重用于线性组合当前层中的所有位置的表示，进而得到上层的第 i 个位置的隐层状态。我们稍后将对其进行正式定义。解码器如图 4.3 右侧所示，它也有 L 层。每层包括三个子层。第一子层是含有掩码的自注意力层，第二子层是解码器-编码器的跨语言注意力子层，第三子层是前馈神经网络子层。在编码器和解码器之中，每个子层都会采用残差连接和层归一化操作。

显然，注意力机制是 Transformer 的关键组成部分。可以将模型中的三种注意力机制（编码器端的自注意力、解码器端的掩码自注意力和编码器-解码器跨语言注意力）形式化为同一个公式：

$$\text{Attention}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{q}\boldsymbol{K}^{\text{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{4.5}$$

其中，$\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}$ 分别表示查询向量、键向量序列和值向量序列。d_k 表示查询向量的维度。

对于编码器端的自注意力机制，其查询向量、键、值向量的来源相同。举例来说，假设我们计算第一层第 j 个位置的输出。用 $\boldsymbol{x}_j$ 表示输入词向量和位置向量的加和向量。那么，查询向量就可以为 $\boldsymbol{x}_j$，键和值向量序列是相同的，都是词向量矩阵 $\boldsymbol{x} = [\boldsymbol{x}_0 \cdots \boldsymbol{x}_n]$。然后，利用多头注意力（假设为 h 个）机制来计算不同子空间中的注意力权重。

$$\begin{aligned}\text{MultiHead}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) &= \text{Concat}(\text{head}_1, \cdots, \text{head}_i, \cdots \text{head}_h)\boldsymbol{W}_{\text{o}} \\ \text{head}_i &= \text{Attention}(\boldsymbol{q}\boldsymbol{W}^i_{\text{Q}}, \boldsymbol{K}\boldsymbol{W}^i_{\text{K}}, \boldsymbol{V}\boldsymbol{W}^i_{\text{V}})\end{aligned} \tag{4.6}$$

其中，Concat 表示将所有注意力头的表示拼接在一起。$\boldsymbol{W}^i_{\text{Q}}, \boldsymbol{W}^i_{\text{K}}$ 和 $\boldsymbol{W}^i_{\text{V}}$ 表示第 i 个头的一组映射矩阵，$\boldsymbol{W}_{\text{o}}$ 表示最终的转换矩阵。

在计算公式 (4.6) 之后，使用残差连接、层正则化以及前馈神经网络，我们可以得到第二层的表示。经过 L 层计算之后，我们可以获得输入词汇序列的上下文表示 $\boldsymbol{C} = [\boldsymbol{h}_0, \cdots, \boldsymbol{h}_n]$。

① 模型和代码详见 https://github.com/tensorflow/tensor2tensor。

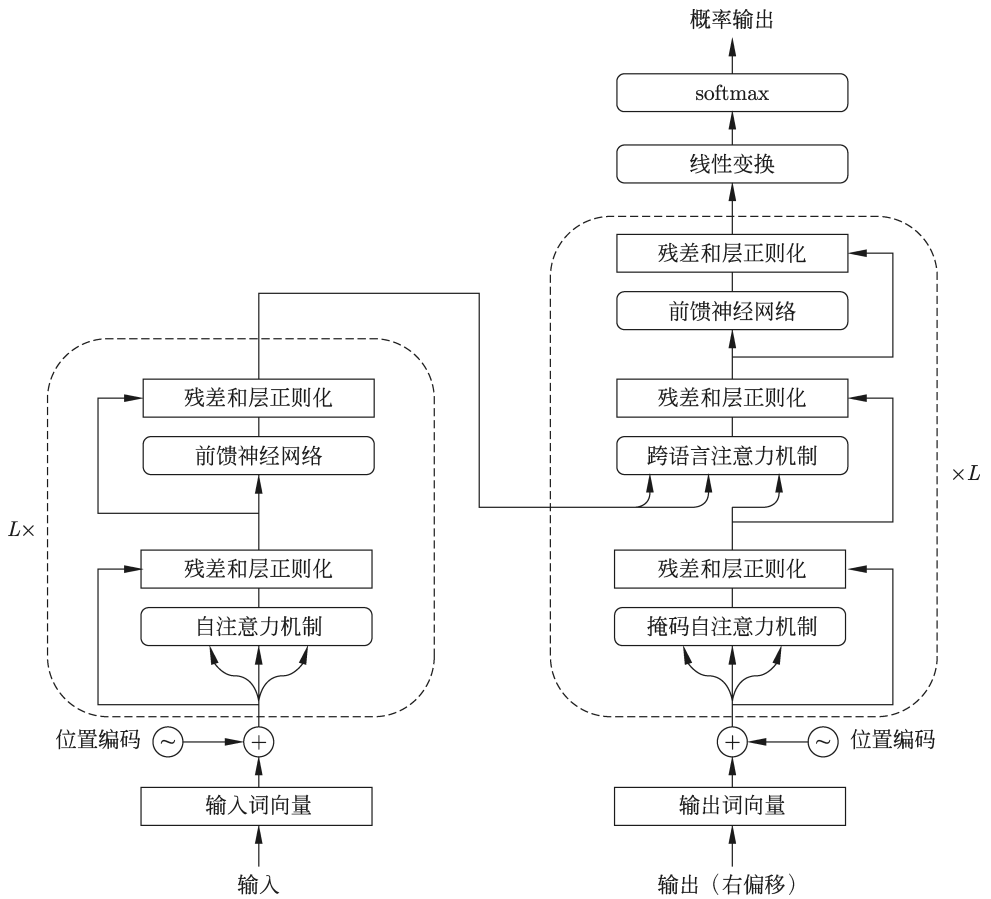


图 4.3 Transformer 模型结构

解码器端的掩码自注意力机制与编码器端的自注意力机制类似，只是第 i 个位置的查询只能关注 i 之前的位置，因为在自左向右的自回归预测过程中，第 i 个位置之后的信息还未生成，因此是不可获取的。

$$\boldsymbol{z}_i = \text{Attention}(\boldsymbol{q}_i, \boldsymbol{K}_{\leqslant i}, \boldsymbol{V}_{\leqslant i}) = \text{softmax}\left(\frac{\boldsymbol{q}_i \boldsymbol{K}_{\leqslant i}}{\sqrt{d_k}}\right)\boldsymbol{V}_{\leqslant i} \tag{4.7}$$

联系编码器和解码器之间的注意力机制需要计算与当前待预测目标端词汇有关的源端动态上下文信息。查询向量是掩码自注意力子层的输出 $\boldsymbol{z}_i$，键和值向量序列都是编码器得到的上下文表示 $\boldsymbol{C}$。残差连接、层正则化和前馈神经网络子层都放置在每层的后面，用于返回整个层的输出表示。经过 L 层计算之后，我们将得到最终的隐层状态 $\boldsymbol{z}_i$。最终，softmax 函数用于计算所有目标语言词汇的概率分布，并根据概率预测结果 y_i，如图 4.3 右上侧所示。

4.3.2 GPT 预训练

如图 4.4 所示，GPT 基于上述单向 Transformer 解码器端的网络结构（不包括编码

器-解码器之间的注意力模块），使用大规模文本数据（如英文书籍语料 BookCorpus[①]）训练一个自左向右的单向语言模型。模型采用掩码自注意力机制来利用所有已经生成的历史文本，同时避免使用尚未产生的未来时刻的文本。正如图 4.4 所阐述的那样，当学习表示 $\boldsymbol{h}_j^1$ 时，x_j 仅仅利用前面的词汇 $\text{SOS}, x_1, \cdots, x_j$。每层使用同样的操作，最终在顶层得到隐层表示 $\boldsymbol{h}_j$。

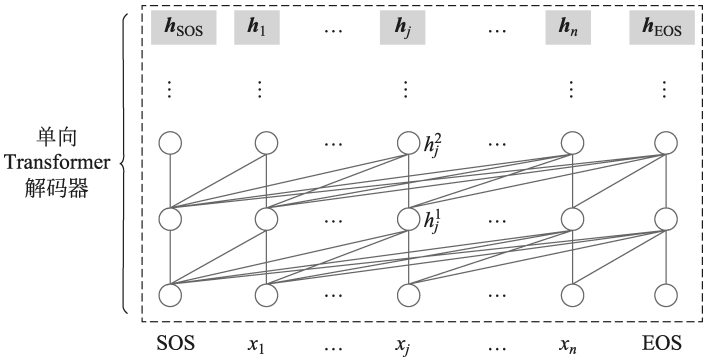


图 4.4 GPT 模型结构

在模型推断阶段，GPT 基于条件概率 $p(x_{j+1} \,|\, x_0, \cdots, x_j)$ 输出下一个时刻的单词 x_{j+1}。在训练阶段，模型参数通过最大化整个训练语料中所有句子的条件最大似然概率值进行优化：

$$L_1 = \sum_{t=1}^{T} \sum_{j=0}^{n+1} \log p\left(x_j^{(t)} \,|\, \text{SOS}, x_1^{(t)}, \cdots, x_{j-1}^{(t)}; \boldsymbol{\theta}\right) \tag{4.8}$$

4.3.3 GPT 微调

在执行下游任务时，预训练的 GPT 常常被作为起点模型，针对具体的目标文本处理任务进行适应性的参数微调。我们知道 GPT 仅以语言模型任务作为目标进行了预训练，因此该模型无法执行诸如文本分类之类的特定任务。因此，有必要使用任务相关的训练数据来微调 GPT 模型以适应相应的应用场景。

假设一个有监督的分类任务包含若干输入序列和输出标签的训练样例，如 (x, y)，其中，$x = (\text{SOS}\ x_1 \cdots x_j \cdots x_n\ \text{EOS})$。预训练好的 GPT 将以 x 为输入，经过 L 层堆叠的掩码自注意力层的计算，产生最上层的表示 $(\boldsymbol{h}_{\text{SOS}}, \boldsymbol{h}_1, \cdots, \boldsymbol{h}_j, \cdots, \boldsymbol{h}_n, \boldsymbol{h}_{\text{EOS}})$。最后，模型使用一个全新的线性输出层和 softmax 函数对 $\boldsymbol{h}_{\text{EOS}}$ 进行分类：

$$p(y \,|\, x) = p(y \,|\, \text{SOS}, x_1, \cdots, x_j, \cdots, x_n, \text{EOS}) = \text{softmax}(\boldsymbol{h}_{\text{EOS}} \boldsymbol{W}_y) \tag{4.9}$$

预训练好的 GPT 模型参数以及线性映射层的参数矩阵 $\boldsymbol{W}_y$ 将会按照最大化似然概率的目标进行微调：

① https://yknzhu.wixsite.com/mbweb。

$$L_2 = \sum_{(x,y)} \log p(y \mid x) \tag{4.10}$$

为了提升泛化能力，同时加速收敛，GPT 在微调过程中将上述两个优化目标进行了组合：

$$L = L_2 + \lambda \times L_1 \tag{4.11}$$

对于那些包含多个序列作为输入的下游任务，GPT 采取了一种简单的方式，即，将这些序列拼接在一起，中间使用分隔符号标记，得到一个长序列从而和预训练的 GPT 模型进行匹配。例如，在文本蕴含任务中，给定条件句 x^1，判断假设句 x^2 是否成立。GPT 使用 $(x^1; \text{Delim}; x^2)$ 作为最终输入序列，其中 Delim 表示分隔符。

Radford 等在 2019 年提出了 GPT 的增强版本 GPT-2[①]，在一系列语言生成任务上取得了出色的性能表现（Radford et al., 2019）。值得注意的是，GPT-2 的模型结构和 GPT 一致。不同点在于 GPT-2 使用了更多的英文文本数据以及更深层次的 Transformer 解码器。其中，英文文本包含了超过 800 万的文档，总量大概为 40GB 的词汇。GPT-2 中最深的模型包含 48 层、约 15.42 亿（1542 million）的网络参数。Radford 等还发现，仅仅使用预训练方式甚至能够在不经过微调的前提下使模型完成下游的语言理解和生成任务（Radford et al., 2019）。例如，他们使用预训练模型生成的摘要质量相当好，达到了一些在 CNN Daily Mail 数据上训练的有监督摘要模型的性能。Brown 等更进一步发明了 GPT-3[②]，其中最大的模型参数量甚至达到了 1750 亿（Brown et al., 2020）。令人惊讶的是，GPT-3 证明只要数据量充足，神经网络模型足够大，那么模型甚至能够在零样本或者少样本的情况下完成大部分自然语言理解和生成的任务。

4.4 BERT：双向 Transformer 编码表示

尽管 GPT 模型在一系列自然语言理解和生成任务上取得了实质性的进展，但是这种自左向右的 GPT 解码器网络学习到的语义表示只能考虑左侧的文本。例如，对于输入词汇 x_j，它的表示仅仅依赖于上文 $x_0, x_1, \cdots, x_{j-1}$，而无法利用下文 $x_{j+1}, \cdots, x_n$。众所周知，左右上下文对于很多文本处理任务都是非常重要的，比如序列标注和自动问答等。因此，Devlin 等在 2019 年提出了全新的预训练与微调模型 BERT[③]，它使用 Transformer 中的双向编码器来充分挖掘上下文信息，以得到更加高效的语义表示（Devlin et al., 2019）。如图 4.5 所示，其中，输入词汇的表示 $\boldsymbol{h}_j$ 通过左侧上文 SOS $x_1, \cdots, x_{j-1}$ 和右侧下文 $x_{j+1}, \cdots, x_n$ EOS 学习得到。

BERT 的贡献主要体现在三个方面。第一，BERT 相比于 GPT 使用了更加深层次的结构，双向编码器包含了至多 24 层的网络，约为 3.4 亿的参数（$\text{BERT}_{\text{LARGE}}$）。第二，

① 代码和模型详见 https://github.com/openai/gpt-2。

② 模型和样例详见 https://github.com/openai/gpt-3。

③ 代码和预训练模型详见 https://github.com/google-research/bert。

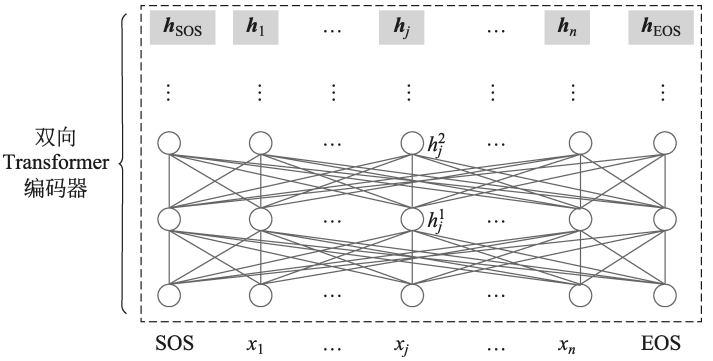


图 4.5 BERT 模型结构

考虑到 BERT 无法按照传统的语言模型进行优化，研究者们设计了两个全新的无监督目标函数，包括掩码语言模型和下一句预测任务。第三，BERT 相比于 GPT 在更大的文本数据集上进行预训练（8 亿词汇的 BookCorpus 和 25 亿词汇的英文维基百科）。BERT 是首个取得重大突破的预训练模型，并且在 11 个自然语言理解任务上达到最佳性能，甚至在自动问答任务中超过了普通人类水平。接下来，我们简单介绍 BERT 预训练和微调的过程。

4.4.1 BERT 预训练

ELMo 和 GPT 都采用条件语言模型作为无监督预训练的优化目标。相比之下，仅利用单向历史上下文进行参数优化的一般语言模型不适用于 BERT，因为 BERT 需要同时利用左右双向的上下文信息，例如，针对特定输入位置，模型需要预测当前位置对应的词汇，而在多层编码器的表示学习过程中该位置的输入词汇将会通过其他位置的传递产生信息泄漏，从而导致该词汇预测自己的问题。我们使用图 4.6 来解释这个问题。假设我们计划使用左侧的上下文 (SOS, x_1) 以及右侧的上下文 $(x_3, \cdots, x_n, \text{EOS})$ 来预测 x_2，即计算概率 $p(x_2 \,|\, \text{SOS}, x_1, x_3, \cdots, x_n, \text{EOS})$。在第一层中，BERT 通过汇聚除了 x_2 自身的全部上下文得到表示 $\boldsymbol{h}_2^1$，如图 4.6 中的灰色虚线所示。在第二层中，BERT 按照同样的方式汇聚全部上下文 $(\boldsymbol{h}_{\text{SOS}}^1, \boldsymbol{h}_1^1, \boldsymbol{h}_3^1, \cdots, \boldsymbol{h}_n^1, \boldsymbol{h}_{\text{EOS}}^1)$ 来学习 $\boldsymbol{h}_2^2$。然而，如图 4.6 中的黑色虚线所示，$\boldsymbol{h}_{\text{SOS}}^1, \boldsymbol{h}_1^1, \boldsymbol{h}_3^1, \cdots, h_n^1$ 和 $\boldsymbol{h}_{\text{EOS}}^1$ 在第一层的表示学习过程中已经包含了 x_2 的信息，因此，$\boldsymbol{h}_2^2$ 将会通过信息传递过程（图 4.6 中的黑色实线）间接地获取 x_2 的信息。所以，使用包含 x_2 的 $\boldsymbol{h}_2^L$ 来预测 x_2 显然是有问题的。

为了解决上述问题，BERT 采用了两种无监督的预测任务来设计预训练的优化目标。其中一个是掩码语言模型，另一个是下一句话预测任务。

掩码语言模型：掩码语言模型的主要方法是按照一定比例对输入序列中的词汇进行随机遮盖，然后通过预测被遮盖的词汇来进行模型的优化。例如，给定输入序列 (SOS x_1 x_2 $\cdots$ x_n EOS)，x_2 可能被随机遮盖掉，这意味着 x_2 将会被一个特殊字符 MASK 代替，如图 4.7 所示。然后，BERT 将会学习遮盖后的词汇序列 (SOS x_1 MASK $\cdots$ x_n EOS) 对应的语义表示，在模型的第 L 层获得最终的隐层表示

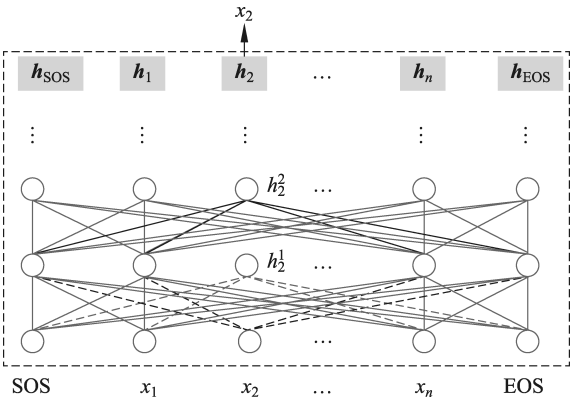


图 4.6 标准的语言模型并不适用于 BERT 训练的原因阐述

$(\boldsymbol{h}_{\text{SOS}}, \boldsymbol{h}_1, \boldsymbol{h}_{\text{MASK}}, \cdots, \boldsymbol{h}_n, \boldsymbol{h}_{\text{EOS}})$。通过对比图 4.7 和图 4.6，我们容易发现由于输入中缺少了 x_2，所以 $\boldsymbol{h}_{\text{MASK}}$ 并不包含 x_2 的相关信息。最终，$\boldsymbol{h}_{\text{MASK}}$ 可以被用来预测 x_2。通过这个例子的阐述，我们可以非常直观地理解掩码语言模型是 BERT 参数优化的合理方法，能够帮助 BERT 更好地利用双向上下文的信息。

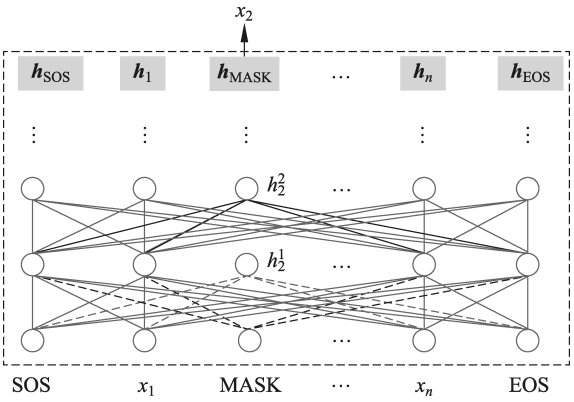


图 4.7 BERT 的掩码语言模型机制阐述

掩码语言模型也将引出一个新的问题，即我们应该遮盖哪些词汇？遮盖多少词汇？实际上，BERT 随机遮盖了每个序列中全部词汇的 15% 并在模型顶层进行预测。可是，因为序列在下游任务执行过程中是未被遮盖的，这就导致了训练和测试不一致的问题。为了解决这个问题，BERT 并没有总是将被遮盖的词汇替换为 MASK。对于这 15% 被选中用于遮盖的词汇，其中的 80% 使用 MASK 替换，10% 使用其他词汇随机替换，另外 10% 保持词汇本身不变。

下一句话预测：有些下游任务的处理对象是两个而非单个词汇序列，如文本蕴涵和问答任务。例如，在文本蕴含任务中，模型需要判决第一个句子（前提句）是否蕴含第二个句子（假设句）。这实际上等价于前提句和假设句拼接在一起并预测“是/否”蕴含的标签。如果 BERT 只是在单个句子上进行预训练，那么它将不适用于这种类型的下游任务。因此，BERT 的设计中还包括另外一个无监督的训练目标，即判断第二个句子 B 是

否自然地承接在第一个句子 A 的后面。例如，文章中有前后两个句子 A 和 B，A 句子是“我来自北京”，B 句子是“北京是中国的首都”。那么 B 是自然承接在 A 后的。如果 B 句子是“总统选举将会在 2020 年举办”，那么，在自然文本中，B 不会直接承接在 A 之后。

训练数据很容易构造。每个预训练的样例 (A B) 可以按照如下策略选取：对于 50% 的训练样例，序列 B 和序列 A 属于同一批单语数据（如 BookCoprus）中的前后两个句子，这些样例视为正样本；同时对于另外 50% 的样例，A 不变，而 B 是随机从语料中选取的句子，这些样例则可以用作负样本。

在预训练过程中，A 和 B 会被拼接成为一个序列 (A[SEP]B)，其中，[SEP] 是分隔两个句子的特殊符号。BERT 学习了该序列 L 层的语义表示，最终第一个词汇[①]的隐层表示 $\boldsymbol{h}_{\text{SOS}}^{L}$ 被输入到一个线性的映射层中，并使用 softmax 层来预测句子 B 是否紧跟在句子 A 的后面。

4.4.2 BERT 微调

与 GPT 相似，预训练好的 BERT 模型可以用来作为下游任务的起点，可以根据目标任务进行简单的适应性参数微调。BERT 仅使用掩码语言模型和下一句话预测作为优化目标，神经网络并不能够直接完成具体下游任务，如文本蕴涵和自动问答。因此，对于这些特定任务，使用 BERT 在任务相关的训练数据上进行微调是必要的。

BERT 主要适用于两种类型的下游任务：分类任务和序列标注任务。对于分类任务，首先将序列输入到预训练好的 BERT 之中，第一个词汇对应的最后一层隐层状态 $\boldsymbol{h}_{\text{SOS}}^{L}$ 将用于后续分类。$\boldsymbol{h}_{\text{SOS}}^{L}$ 将被参数矩阵 $\boldsymbol{W}_{\text{o}}$ 线性映射之后送入 softmax 层中计算目标类别的概率分布。预训练好的 BERT 网络参数以及新引入的映射矩阵 $\boldsymbol{W}_{\text{o}}$ 将会在分类任务的训练集上以最大化概率 $p(y \,|\, x)$（y 是标签）为目标进行微调。

对于序列标注任务，每一个词汇 x_j 通过预训练的 BERT 可以得到一个最后一层的隐层表示 $\boldsymbol{h}_j^L$。之后 $\boldsymbol{h}_j^L$ 通过线性映射以及 softmax 层来预测标签 y_j。所有的网络参数会在序列标注的训练数据上以最大化概率 $p(y \,|\, x)$（y 是标签序列）为目标进行微调。

4.4.3 XLNet: 广义自回归预训练模型

尽管已经在很多文本处理任务中取得了成功，BERT 仍然存在很多不足之处。最关键的是 BERT 的预训练和微调之间仍然存在着严重的不匹配问题，因为在预训练过程中大量使用的特殊符号 MASK 几乎不会在下游任务的微调过程中出现。此外，BERT 假设输入序列中遮盖的词汇之间互相独立。依据 BERT 的设定，15% 的输入词汇将被随机遮盖。举例来说，原始的输入序列 $(\text{SOS}, x_1, x_2, \cdots, x_{j-1}, x_j, x_{j+1}, \cdots, x_{n-1}, x_n, \text{EOS})$ 在随机遮盖之后可能变为 $(\text{SOS}, x_1, \text{MASK}, \cdots, x_{j-1}, \text{MASK}, x_{j+1}, \cdots, \text{MASK}, x_n, \text{EOS})$。

① 我们在介绍中使用 SOS 表示第一个词汇，而在 BERT 论文中采用的是另一个特别符号 [CLS]，实际效果是一样的。

显然，在 BERT 的预训练过程中，x_2 和 x_{n-1} 将不会用于 x_j 的预测，类似地，(x_2, x_j) 也不会对 x_{n-1} 的预测起作用，(x_j, x_{n-1}) 对 x_2 同理。实际上，x_2，x_j 和 x_{n-1} 之间可能存在互相依赖的关系。

为了克服上述问题，Yang 等在 2019 年提出了一种广义的自回归预训练模型 XLNet[①]（Yang et al., 2019）。该模型旨在保持 BERT 利用双向上下文这个优点的同时，不再使用掩码遮盖机制。XLNet 相比于 BERT 主要有两个创新想法：排列语言模型和两路自注意力机制。

排列语言模型：直观地，如果我们枚举输入序列全部的排列形式，那么所有的上下文均有机会出现在焦点词 x_j 之前。使用序列 (x_1, x_2, x_3, x_4) 举例，假设 x_3 是焦点词。如图 4.8 所示，不同的排列将会为 x_3 提供不同的上下文。右下角的排列将所有左右双向的上下文移动到了 x_3 的前方。因此，XLNet 预训练模型可以基于任何自回归语言模型进行参数优化。

假设 $\boldsymbol{Z}_n$ 表示包含 n 个词汇的序列对应的所有可能排列的集合，$z_j, z_{<j}$ 分别表示特定排列 $z \in \boldsymbol{Z}_n$ 中的第 j 个元素和前 $j-1$ 个元素。基于上述设置，XLNet 将会按照最大化排列集合中自回归语言模型概率的期望来进行预训练。

$$\sum_{t=1}^{T} \left\{ E_{\boldsymbol{z} \in \boldsymbol{Z}_n} \left[\sum_{j=1}^{n} \log p(x_{z_j}^{(t)} \mid x_{z_{<j}}^{(t)}; \boldsymbol{\theta}) \right] \right\} \tag{4.12}$$

值得注意的是，XLNet 仅置换分解顺序（用于计算概率 $p(x)$ 的分解方法），而不是对原始序列进行重新排序，如图 4.8 所示。

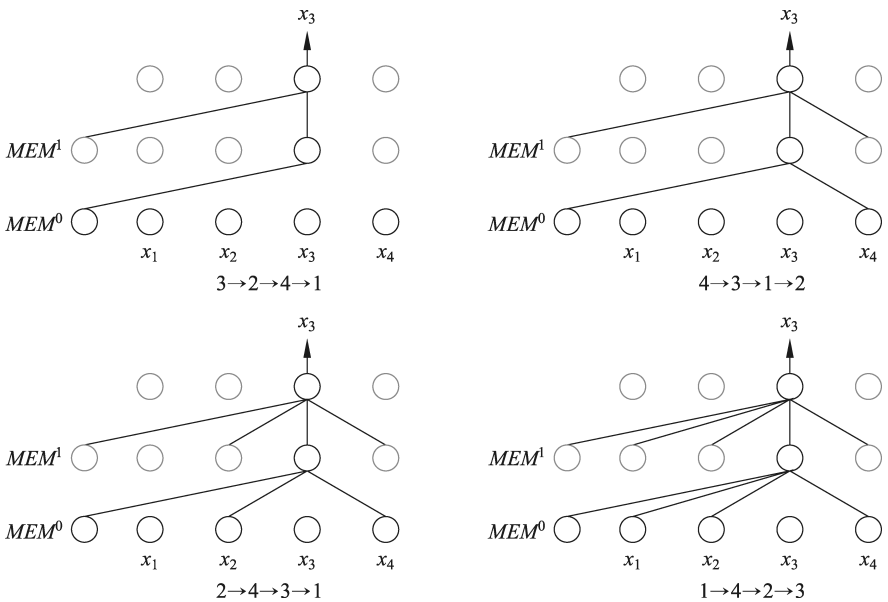


图 4.8 序列 (x_1, x_2, x_3, x_4) 的排列以及相应的自回归语言模型

① 代码和预训练模型见 https://github.com/zihangdai/xlnet。

双路自注意力机制：在排列语言模型中计算 $p(x_{z_j} \mid x_{z_{<j}})$ 时，隐层状态 $\boldsymbol{h}(x_{z_j})$ 是通过 Transformer 自注意力机制学习得到的，softmax 函数后续被用来计算下一个词汇的概率分布。容易发现使用 $\boldsymbol{h}(x_{z_{<j}})$ 预测 x_j 时并不知晓目标位置 j。因此，$p(x_{z_j} \mid x_{z_{<j}}; \theta)$ 的计算与目标位置无关，与 $p(x_{z_k} \mid x_{z_{<j}}; \theta)(k \geqslant j)$ 共享了相同的概率分布。这意味着，在给定相同历史上下文的条件下，某个词汇在条件语言模型下的概率与位置无关。显然，这种位置不敏感的特性是我们不希望出现的，因为语言本身和词汇的顺序与出现位置紧密相关。因此，XLNet 设计了全新的双路自注意力机制来解决该问题。

模型将会在第 j 个时刻学习两种隐层表示：内容表示 $\boldsymbol{h}(x_{z_{\leqslant j}})$ 以及查询表示 $\boldsymbol{g}(x_{z_{<j}})$。

$$\boldsymbol{h}^l(x_{z_j}) = \text{Attention}(\boldsymbol{q}_j = \boldsymbol{h}^{l-1}(x_{z_j}), \boldsymbol{K}_{\leqslant j}\boldsymbol{V}_{\leqslant j} = \boldsymbol{h}^{l-1}(x_{z_{\leqslant j}})) \tag{4.13}$$

$$\boldsymbol{g}^l(x_{z_j}) = \text{Attention}(\boldsymbol{q}_j = \boldsymbol{g}^{l-1}(x_{z_j}), \boldsymbol{K}_{\leqslant j}\boldsymbol{V}_{\leqslant j} = \boldsymbol{h}^{l-1}(x_{z_{<j}})) \tag{4.14}$$

注意，内容表示 $\boldsymbol{h}(x_{z_{\leqslant j}})$ 与传统 Transformer 中的隐层表示一致。查询表示 $\boldsymbol{g}(x_{z_{<j}})$ 是位置相关的，但是它的学习过程不使用第 z_j 个词汇的内容信息。顶层的查询表示 $\boldsymbol{g}^l(x_{z_{<j}})$ 将会被用于预测 x_{z_j}。最初，$\boldsymbol{h}^0(x_{z_j})$ 是词汇 x_{z_j} 的词向量表示，$\boldsymbol{g}^0(x_{z_j})$ 是可训练的向量 $\boldsymbol{w}$。图 4.9 阐明了双路自注意力机制的主要思想。假设序列分解顺序为 $2 \to 4 \to 3 \to 1$，我们需要在给定 $(\boldsymbol{x}_2, \boldsymbol{x}_4)$ 的条件下预测 $\boldsymbol{x}_3$。灰色实线代表内容表示的学习过程（与传统 Transformer 一致）。黑色实线代表查询表示的学习过程。黑色虚线表示该输入仅仅被用来作为查询，它的内容值在注意力计算过程中并不被使用。例如，$\boldsymbol{g}_3^1$ 是词向量 $\boldsymbol{x}_2$ 和 $\boldsymbol{x}_4$ 的加权和（$\boldsymbol{x}_3$ 被排除在外）。权重使用 $\boldsymbol{g}_3^0 = \boldsymbol{w}$ 作为查询向量与 $\boldsymbol{x}_2$ 和 $\boldsymbol{x}_4$ 进行计算。如果 XLNet 仅包含两层，那么 $\boldsymbol{g}_3^2$ 将会被用来预测 $\boldsymbol{x}_3$。

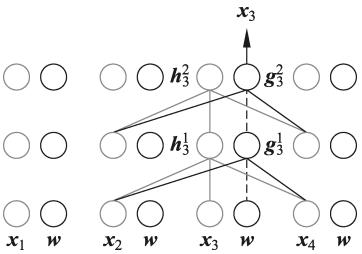


图 4.9 对于 $2 \to 4 \to 3 \to 1$ 的双路注意力模型

为了加快训练过程的收敛速度，XLNet 仅仅预测每个被采样到的因式分解样本下的最后几个词汇而非整个序列。此外，XLNet 还整合了一些复杂的技术，例如，Transformer-XL 的相对位置编码和分段递归机制（Dai et al., 2019）。最终，XLNet 在 20 个文本处理任务上取得了优于 BERT 的性能。

有趣的是，Facebook 的研究者（Liu et al., 2019）发现 BERT 远远没有训练充分。他们报告称，通过对关键参数以及训练数据规模的细致设计，BERT 模型[①]能够匹敌甚至超越 XLNet 以及其他变体。

① 他们将实现的结果命名为 RoBERTa，具体细节见 https://github.com/pytorch/fairseq。

4.4.4 UniLM

ELMo，BERT 和 XLNet 旨在全面探索输入序列的双向上下文信息，并主要用于自然语言理解任务。GPT 适用于自然语言生成任务，如摘要等。尽管如此，GPT 只能利用左侧上下文信息。一个有趣的问题是能否结合 BERT 和 GPT 各自的优势来设计用于文本生成任务的预训练模型。

Dong 等在 2019 年提出一种统一的预训练语言模型 UniLM[①]，它可以适应性地将 Transformer 模型用于单语言的序列到序列任务（Dong et al., 2019）。给定单语语料中两个连续的序列 (x, y)，UniLM 认为 x 是输入序列，y 是输出序列。如图 4.10 所示，UniLM 使用双向 Transformer 编码器来接收输入 x，使用单向 Transformer 解码器来生成 y。UniLM 采用了与 BERT 类似的掩码机制，即通过设置一个掩码矩阵动态地实现每个时刻可访问的上下文信息。通过对大规模单语数据的预训练，UniLM 后续可以通过微调来完成文本生成任务，如摘要和问题生成。根据 Dong 等的报告，UniLM 在 CNN Daily Mail 数据集上摘要的性能达到最佳。

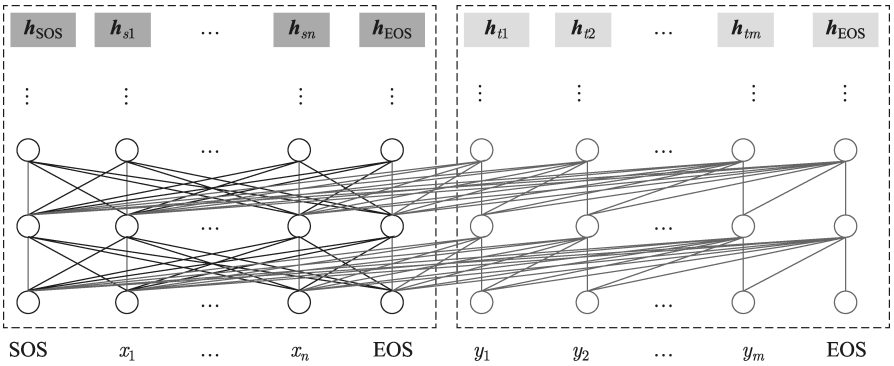


图 4.10 UniLM 模型结构

为了更好地理解各类预训练语言模型，我们将各个模型列在表 4.1 中，并简述它们的关键结构和特征。

表 4.1 不同预训练语言模型的比较

模型	结构	关键特征	最适用的任务
ELMo	双向 LSTMs	首个可以获取动态词向量的大规模预训练模型	文本理解
GPT	Transformer 解码器	预训练和微调采用统一模型的首个方法	文本理解
GPT-2	Transformer 解码器	使用了更多的数据、更深的网络	文本生成
GPT-3	Transformer 解码器	使用特别多的数据、特别深的网络	文本生成
BERT	Transformer 编码器	掩码语言模型和下一句话预测任务为优化目标的去噪自编码器	文本理解
XLNet	Transformer 编码器	基于序列排列组合泛化的自回归语言模型	文本理解
UniLM	Transformer	基于泛化掩码机制的预训练模型，能够同时处理理解和生成任务	文本生成

① 代码和模型详见 https://github.com/microsoft/unilm。

4.5 进一步阅读

本章简要介绍了几种流行的预训练模型，包括 ELMo，GPT，BERT，XLNet 和 UniLM。我们可以看到，预训练和微调范式已在许多自然语言理解和生成任务上取得了重大突破。最近，预训练方法发展迅速，许多改进的模型相继提出，其中大多数集中在 BERT 框架的改进上。这些新的模型可以大致分为以下三类[①]。

其中的一个研究方向旨在设计更复杂的目标函数或将知识整合到 BERT 架构中。Sun 等提出了模型 ERNIE，该模型通过遮盖实体而不是像 BERT 那样遮盖字符（子词或者单词）来改进掩码语言模型（Sun et al., 2019）。他们证明了实体掩码模型在许多中文文本处理任务中都有很好的性能表现。他们进一步将该模型升级到 ERNIE-2.0，该模型使用多任务学习框架逐步学习预训练任务（Sun et al., 2020）。Zhang 等在 BERT 的基础上提出了另一个改进的模型，也称为 ERNIE，该模型将知识图谱中实体的表示学习纳入了 BERT 预训练过程之中（Zhang et al., 2019）。

另一个研究方向旨在尽可能地压缩预训练语言模型。由于 BERT 非常繁重且包含大量参数，因此它在计算上非常昂贵且占用大量内存，尤其是对于后续推理步骤而言。（Sanh et al., 2019）、（Tang et al.,2019）和（Jiao et al., 2020）提出使用知识蒸馏的策略，在性能下降可以忽略不计的前提下将大模型压缩为小模型。Lan 等提出通过两种参数削减方法来减少内存使用并加快 BERT 的训练过程，即参数化向量的因式分解和跨层参数的共享策略（Lan et al., 2020）。

第三个方向探讨了预训练模型的生成任务和跨语言任务。尽管大多数研究通过增强 BERT 来解决自然语言理解任务，但是越来越多的研究人员将注意力转移到生成任务和跨语言任务的预训练上。除了 UniLM 设计了面向生成的预训练模型，Song 等提出的 MASS 模型（Song et al., 2019）、Lewis 等提出的 BART 模型（Lewis et al., 2020）和 Raffel 等提出的 T5 模型（Raffel et al., 2020）促进了生成式预训练模型的进步。以 MASS 模型为例，该模型对句子中的连续子序列 seq 进行遮盖，并使用被遮盖的序列作为输入，之后通过序列到序列的模型来预测被遮盖的连续子序列 seq。跨语言预训练也吸引了越来越多研究者的注意。Lample 和 Conneau 提出了 XLM 模型，该模型使用平行句对作为输入来进行跨语言模型的预训练（Lample and Conneau, 2019）。

另一个值得注意的问题是文本生成任务中的推理过程几乎都遵循从左到右的解码方式，无法利用未来信息。一个有希望的研究方向是对生成任务执行同步双向推理，例如，机器翻译中的类似工作（Zhou et al., 2019）。

习 题

4.1 假设 GPT 和 BERT 模型只考虑自注意力的计算、ELMo 仅采用简单的循环神经网络计算单元，请对比三个预训练模型 ELMo、GPT 和 BERT 的时间复杂度。

① 有关预训练模型的更多信息，请参见 Qiu 等人在 2020 年的综述报告。

4.2 请阐述为什么基于掩码语言模型的 BERT 实际上就是一种去噪自编码器。

4.3 经验表明，BERT 更擅长于理解式任务，GPT 更擅长于生成式任务，BART 被认为同时擅长理解式和生成式任务，请查阅相关文献说明 BART 完成理解式任务和生成式任务的方式，并从目标函数设计的角度说明为什么 BART 可以同时处理生成式和理解式任务。

4.4 BERT 采用掩码语言模型和下一句话预测作为优化目标函数，请分析下一句话预测这个目标函数的作用，说明该目标函数会提升哪些任务的性能，并给出理由。

4.5 MASS、BART 和 T5 都是序列到序列的生成模型，请查看文献后阐述三个模型之间的异同，说明各模型最适合的下游任务类型，并给出原因。

第 5 章　文 本 分 类

5.1　概　　述

文本分类是按照一定的分类体系对文本类别进行自动标注的过程。其目标是在给定分类体系下，将文本集中的每个文本划分到某个或者某几个类别中，如图 5.1 所示。常见的文本分类任务包括文本主题分类、体裁分类、垃圾邮件识别等。

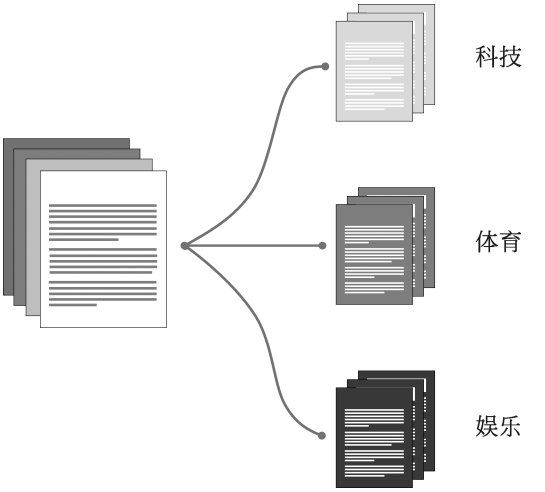


图 5.1　文本分类任务示例

早期的文本分类方法以规则方法为主，但是这种方法往往需要专家精心制定分类规则，规则集的建立和维护都非常耗时耗力。20 世纪 90 年代以后，随着统计机器学习算法的兴起，基于监督机器学习的分类算法在文本分类任务中取得了很大的成功。常见的文本分类算法包括：朴素贝叶斯（naïve Bayes, NB）、logistic 回归、最大熵（maximum entropy, ME）模型和支持向量机（support vector machine, SVM）等。近年来，以卷积神经网络和循环神经网络为代表的深度神经网络技术在文本分类任务上都取得了较大的进展，逐渐发展成为当下研究中的主流方法。

基于传统机器学习方法的文本分类系统可以示意性地用图 5.2 表示，它主要由文本表示、特征选择（feature selection）、分类器设计三部分组成，文献（Sebastiani, 2002）按照这一基本结构对文本分类技术进行了综述。本章首先遵循这一结构顺序介绍基于传

统机器学习方法的文本分类方法，然后单独介绍基于深度神经网络的文本分类方法，最后介绍文本分类中的性能评估方法。

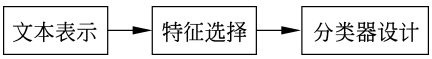


图 5.2 基于传统机器学习的文本分类系统框架

5.2 传统文本表示

在文本分类任务中，如何准确、高效地表示一个文本对于后续的分类算法非常重要。一方面，要求表示方法能够真实地反映文本的内容；另一方面，又要求该方法对不同类型的文本有足够的区分能力。本书第 3 章已经介绍了常见的文本表示方法，这里不再赘述。但需要进一步说明的是，对于不同的分类模型，其相应的文本表示方法也有所不同。如传统的线性分类模型（如 logistic 回归、线性支持向量机）通常以向量空间模型进行文本表示，而生成式模型（generative model）的文本表示则是由类条件分布假设确定，如在朴素贝叶斯模型中多项分布（multinormial distribution）假设对应的是词袋模型（词袋模型与向量空间模型是类似的，但它不支持实数值特征）。

用向量空间模型进行文本表示需要经过以下两个主要步骤：一是根据训练集生成文本特征序列，二是依据特征序列对训练文本集和测试样本集中的各个文档进行赋权值和规范化等处理，将其转化为机器学习算法所需的特征向量。需要注意的是，向量空间模型虽然简单、高效，但是它丢失了原始文档的很多信息，因此，为了提高文本分类的性能，往往需要借助特征工程向特征空间中引入更多的语言学特征，如 n 元词序信息、句法信息和语义信息等。另外，对于不同的文本分类任务，甚至对于不同的语料，所采用的最优特征权重方法也有所不同。如在文档主题分类任务中，TF-IDF 权重常常效果最好，而在文本情感分类任务中，Bool 权重则得到了更加广泛的使用。

表 5.1 给出了一个文本分类数据集，该数据集的类别包括“教育”和“体育”两个类别，训练集中每个类别各有两个文档，测试集一共包括两个文档。表 5.2 给出了该数据集对应的词表，每个文档可以表示为以词表作为基的向量空间中的一个向量。

表 5.1 文本分类数据集

序号	文　　档	类别
$train\_d_1$	北京 理工 大学 计算机 专业 创建 于 1958 年 是 中国 最早 设立 计算机 专业 的 高校 之一	教育
$train\_d_2$	北京 理工 大学 学子 在 第四 届 中国 计算机 博弈 锦标赛 中 夺冠	教育
$train\_d_3$	北京 理工 大学 体育馆 是 2008 年 中国 北京 奥林匹克 运动会 的 排球 预赛 场地	体育
$train\_d_4$	第五 届 东亚 运动会 中国 军团 奖牌 总数 创 新高 男女 排球 双双 夺冠	体育
$test\_d_1$	北京 理工 大学 是 理工 为主 工理文 协调 发展 的 全国 重点 大学	
$test\_d_2$	复旦 大学 排球 队 获得 本届 大学生 运动会 排球 比赛 冠军	

表 5.2 文本分类数据集（表 5.1）对应的词表

奥林匹克	北京	博弈	场地	创	创建	大学	第四	第五	东亚	夺冠	高校	计算机	奖牌	届
锦标赛	军团	理工	男女	年	排球	设立	双双	体育馆	新高	学子	预赛	运动会	之一	中
中国	专业	总数	最早											

5.3 特 征 选 择

传统的向量空间模型基于高维稀疏的向量表示文本，因此，在进行分类算法之前通常需要对高维的特征空间进行降维。降维方法主要分为两类：特征提取（feature extraction）和特征选择（feature selection）。

特征提取的目的是将原始的高维稀疏特征空间映射为低维稠密的特征空间。在模式识别领域，经典的特征提取方法有主成分分析（principal component analysis, PCA）方法和独立成分分析（independent component analysis，ICA）方法等，但是这些方法在文本分类中并不常用。曾有学者基于潜在语义索引（latent semantic indexing, LSI）进行文本降维，该方法使用文本的主题特征代替传统特征，降维作用显著，但是单独使用主题特征往往效果一般。实际上，在自然语言处理领域 LSI 与 PCA 属于同源的方法，其本质都是进行奇异值分解（singular value decomposition, SVD）。此外，概率潜在语义分析（probabilistic latent semantic analysis, PLSA）和潜在狄利克雷分布（latent Dirichlet allocation，LDA） 模型也曾被应用于文本分类特征降维，但是因效率和效果欠佳都未获得大规模的应用。

特征选择是从特征空间中择优选出一部分特征子集的过程。文本分类领域常见的特征选择方法包括无监督特征选择和有监督特征选择两类。前者可以应用于没有类别标注的语料（如文本聚类），但是效果往往较差，常见方法包括基于词频 TF（或者文档频率 DF）的特征选择。后者依赖于类别标注信息，可以有效地针对分类问题选择出较优的特征子集，常见方法包括互信息法（MI）、信息增益法（IG）和卡方统计量法（χ^2）等。文献（Yang and Pedersen, 1997）和（Forman, 2003）总结了文本分类中的特征选择方法并指出，一个好的特征选择算法可以有效地对特征空间进行降维，提高分类器的效率，同时去除冗余特征和噪声特征，提高文本分类的性能。

本节主要介绍文本分类中的有监督特征选择方法。

5.3.1 互信息法

在信息论中，假设 X 是一个离散型随机变量，其概率分布为 $p(x) = P(X = x)$，那么，X 的熵（entropy） $H(X)$ 定义为

$$H(X) = -\sum_{x} p(x) \log p(x) \tag{5.1}$$

熵用于度量随机变量的不确定性。一个随机变量的熵越大，其不确定性越大，表示该变量需要的信息量越大；反之，熵越小，则不确定性越小，表示该变量需要的信息量也越小。

假设 X 和 Y 是一对随机变量，服从联合分布 $p(x,y)=P(X=x,Y=y)$，那么，X,Y 的联合熵（joint entropy）定义为

$$H(X,Y)=-\sum_{x}\sum_{y}p(x,y)\log p(x,y) \tag{5.2}$$

联合熵描述的是刻画一对随机变量需要的信息量。

条件熵（conditional entropy）描述的是在已知随机变量 X 取值的前提下，随机变量 Y 的不确定性程度。或者说，在已知 X 取值的条件下，表示 Y 还需要的额外信息量。定义如下：

$$\begin{aligned}H(Y|X)&=\sum_{x}p(x)H(Y|X=x)\\&=-\sum_{x}\sum_{y}p(x,y)\log p(y|x)\end{aligned} \tag{5.3}$$

当且仅当 Y 的值完全由 X 确定时，$H(Y|X)=0$；反之，当且仅当 Y 和 X 相互独立时，$H(Y|X)=H(Y)$。

熵、联合熵和条件熵的关系为

$$H(Y|X)=H(X,Y)-H(X) \tag{5.4}$$

图 5.3 描述了上述各信息量之间的关系。左侧的圆形表示熵 $H(X)$，右侧的圆形表示熵 $H(Y)$，两个圆形的并集表示联合熵 $H(X,Y)$，左侧的月牙形表示条件熵 $H(X|Y)$，右侧的月牙形表示条件熵 $H(Y|X)$。那么，两个圆形的交集表示什么呢？这就是我们下面要引入的互信息（mutual information, MI）$I(X;Y)$。

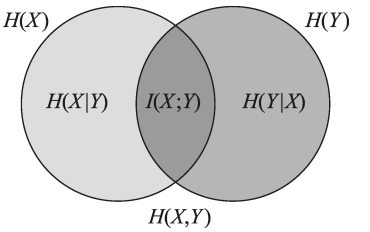


图 5.3 熵、联合熵、条件熵与互信息的关系

互信息反映的是两个随机变量相互关联的程度。对于离散随机变量 X 和 Y，其互信息定义为

$$I(X;Y)=\sum_{x,y}p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \tag{5.5}$$

熵、条件熵和互信息之间存在如下关系：

$$I(X;Y)=H(Y)-H(Y|X)=H(X)-H(X|Y) \tag{5.6}$$

两个随机变量的互信息是变量间相互依赖性的量度，它可以看成是一个随机变量中包含的关于另一个随机变量的信息量，或者说是一个随机变量由于已知另一个随机变量而减少的不确定性。

将 $I(x;y) = \log \dfrac{p(x,y)}{p(x)p(y)}$ 记为随机变量 (X,Y) 取确定值 (x,y) 时的点式互信息（pointwise mutual information，PMI）。由式（5.5）可以看出，互信息是点式互信息的期望。在文本分类中，通常用点式互信息衡量特征项 t_i 透露类别 c_j 的信息量。

对于给定的语料，首先针对每个特征 t_i 和每个类别 c_j，统计表 5.3 中的数值。表中的 N_{t_i,c_j} 表示特征项 t_i 在第 c_j 类文档中出现的文档频率，$N_{t_i,\bar{c}_j}$ 表示特征项 t_i 在所有非第 c_j 类文档中出现的文档频率，$N_{\bar{t}_i,c_j}$ 表示 t_i 以外的所有特征项在第 c_j 类文档中出现的文档频率，$N_{\bar{t}_i,\bar{c}_j}$ 表示 t_i 以外的所有特征项在所有非第 c_j 类文档中出现的文档频率，$N = N_{t_i,c_j} + N_{t_i,\bar{c}_j} + N_{\bar{t}_i,c_j} + N_{\bar{t}_i,\bar{c}_j}$ 表示文档总数。

表 5.3 按特征和类别统计的文档频率

特征	类别	
	c_j	$\bar{c}_j$
t_i	N_{t_i,c_j}	$N_{t_i,\bar{c}_j}$
$\bar{t}_i$	$N_{\bar{t}_i,c_j}$	$N_{\bar{t}_i,\bar{c}_j}$

然后，根据最大似然估计原理，用频率估计以下概率：

$$p(c_j) = \frac{N_{t_i,c_j} + N_{\bar{t}_i,c_j}}{N} \tag{5.7}$$

$$p(t_i) = \frac{N_{t_i,c_j} + N_{t_i,\bar{c}_j}}{N} \tag{5.8}$$

$$p(c_j|t_i) = \frac{N_{t_i,c_j}}{N_{t_i,c_j} + N_{t_i,\bar{c}_j}} \tag{5.9}$$

$$p(c_j|\bar{t}_i) = \frac{N_{\bar{t}_i,c_j}}{N_{\bar{t}_i,c_j} + N_{\bar{t}_i,\bar{c}_j}} \tag{5.10}$$

为了防止出现零概率事件，$p(c_j|t_i)$ 和 $p(c_j|\bar{t}_i)$ 的估计可使用拉普拉斯平滑（Laplace smoothing）（分子加 1，分母加上类别数 M）。

那么，t_i 和 c_j 之间的互信息 $I(t_i;c_j)$ 可以计算为

$$I(t_i;c_j) = \log \frac{N_{t_i,c_j} N}{\left(N_{t_i,c_j} + N_{\bar{t}_i,c_j}\right)\left(N_{t_i,c_j} + N_{t_i,\bar{c}_j}\right)} \tag{5.11}$$

为了衡量特征项 t_i 对于全部类别的信息量，可以对各类按概率加权平均（也可以理解为特征项 t_i 与类别随机变量 C 的互信息）：

$$I_{\text{avg}}(t_i) = \sum_j p(c_j)\, I(t_i;c_j) \tag{5.12}$$

或者取各类中的最大值

$$I_{\max}(t_i) = \max_j \{I(t_i; c_j)\} \tag{5.13}$$

作为该特征的互信息值。

特征选择的过程是指对全部的特征项计算互信息值，按照得分进行排序，最终选择排在前面的一部分特征作为优选的特征子集。表 5.4 给出了用 MI 法对文本分类数据集（表 5.1）进行特征选择的结果。

表 5.4 用 MI 法对文本分类数据集（表 5.1）进行特征选择的结果

特　　征	MI
计算机 排球 运动会	0.4055
1958 2008 奥林匹克 博弈 场地 创 创建 第四 第五 东亚 高校 奖牌 锦标赛 军团 男女 设立 双双 体育馆 新高 学子 于 预赛 在 之一 中 专业 总数 最早	0.2877
北京 大学 理工	0.1823
的 夺冠 届 年 是 中国	0.0000

5.3.2 信息增益法

信息增益（information gain，IG）是指在给定随机变量 X 的条件下，随机变量 Y 的不确定性减少的程度：

$$G(Y|X) = H(Y) - H(Y|X) \tag{5.14}$$

这种减少的程度用 Y 的熵 $H(Y)$ 与条件熵 $H(Y|X)$ 之间的差值表示。与式 (5.6) 比较，可以发现信息增益和互信息是等价的。互信息可以看成是随机变量 Y 中包含的关于随机变量 X 的信息量，也可以理解为随机变量 Y 由于随机变量 X 已知而减少的不确定性（即信息增益）。值得提及的是，在文本分类特征选择方法中，互信息法的实质是点式互信息法，而信息增益法的实质是互信息法。

将特征项 $T_i \in \{t_i, \bar{t}_i\}$ 看作一个服从伯努利分布（Bernoulli distribution, 也称 0-1 分布）的二元随机变量，同时将类别 C 视为服从类别分布（categorical distribution）的随机变量，那么，信息增益定义为熵 $H(C)$ 与条件熵 $H(C|T_i)$ 的差值：

$$\begin{aligned} G(T_i) &= H(C) - H(C|T_i) \\ &= -\sum_j p(c_j) \log p(c_j) - \left[\left(-\sum_j p(c_j, t_i) \log p(c_j|t_i) \right) + \right. \\ &\quad \left. \left(-\sum_j p(c_j, \bar{t}_i) \log p(c_j|\bar{t}_i) \right) \right] \end{aligned} \tag{5.15}$$

信息增益考虑了 $\{t_i, \bar{t}_i\}$ 两种情形，因此可以写成互信息 $I(t_i; c_j)$ 和 $I(\bar{t}_i; c_j)$ 的加权平均（Yang and Pedersen, 1997）：

$$G(T_i) = \sum_j p(t_i, c_j) I(t_i; c_j) + p(\bar{t}_i, c_j) I(\bar{t}_i; c_j) \tag{5.16}$$

总的来说，IG 法进行文本分类特征选择的效果比 MI 法更好。

表 5.5 给出了用 IG 法对文本分类数据集（表 5.1）进行特征选择的结果。

表 5.5 用 IG 法对文本分类数据集（表 5.1）进行特征选择的结果

特 征	IG
计算机 排球 运动会	0.1308
1958 2008 奥林匹克 博弈 场地 创 创建 第四 第五 东亚 高校 奖牌 锦标赛 军团 男女	0.0293
设立 双双 体育馆 新高 学子 于 预赛 在 之一 中 专业 总数 最早 北京 大学 理工	
的 夺冠 届 年 是 中国	0.0000

5.3.3 卡方统计量法

卡方 (χ^2) 检验是以分布为基础的一种假设检验方法，其基本思想是通过计算观察值与期望值的偏差确定假设是否成立。卡方检验常用于检测两个随机变量的独立性。

在特征选择中，定义特征项 $T_i \in \{t_i, \bar{t}_i\}$ 和类别 $C_j \in \{c_j, \bar{c}_j\}$ 分别为服从伯努利分布的二元随机变量，t_i 和 $\bar{t}_i$ 分别表示特征项 t_i 出现和不出现，c_j 和 $\bar{c}_j$ 分别表示文档类别是否为 c_j。

首先提出原假设：T_i 和 C_j 相互独立，即 $p(T_i, C_j) = p(T_i)\,p(C_j)$。对于每个特征项 T_i 和每个类别 C_j，计算如下统计量：

$$\chi^2(T_i, C_j) = \sum_{T_i \in \{t_i, \bar{t}_i\}} \sum_{C_j \in \{c_j, \bar{c}_j\}} \frac{(N_{T_i,C_j} - E_{T_i,C_j})^2}{E_{T_i,C_j}} \tag{5.17}$$

其中，N 是观察频率，E 是符合原假设的期望频率。例如，N_{t_i,c_j} 是基于样本集观测得到的特征项 t_i 出现在第 c_j 类文档中的文档频率，E_{t_i,c_j} 是指在原假设成立条件下的特征项 t_i 出现在第 c_j 类文档中的文档频率。用表 5.3 的统计，E_{t_i,c_j} 的计算如下：

$$\begin{aligned} E_{t_i,c_j} &= N \cdot p(t_i, c_j) = N \cdot p(t_i) \cdot p(c_j) \\ &= N \cdot \frac{N_{t_i,c_j} + N_{t_i,\bar{c}_j}}{N} \cdot \frac{N_{t_i,c_j} + N_{\bar{t}_i,c_j}}{N} \end{aligned} \tag{5.18}$$

类似地计算 $E_{\bar{t}_i,c_j}$，$E_{t_i,\bar{c}_j}$ 和 $E_{\bar{t}_i,\bar{c}_j}$，代入式 (5.17)，得到如下卡方统计量（χ^2 statistic）的算式：

$$\chi^2(T_i, C_j) = \frac{N \cdot \left(N_{t_i,c_j} N_{\bar{t}_i,\bar{c}_j} - N_{\bar{t}_i,c_j} N_{t_i,\bar{c}_j}\right)^2}{\left(N_{t_i,c_j} + N_{\bar{t}_i,c_j}\right) \cdot \left(N_{t_i,c_j} + N_{t_i,\bar{c}_j}\right) \cdot \left(N_{t_i,\bar{c}_j} + N_{\bar{t}_i,\bar{c}_j}\right) \cdot \left(N_{\bar{t}_i,c_j} + N_{\bar{t}_i,\bar{c}_j}\right)} \tag{5.19}$$

$\chi^2(T_i, C_j)$ 值越高，说明 T_i 与 C_j 之间的独立假设越不成立，它们的相关性越高。

同样地，对 $\chi^2(T_i, C_j)$ 按照各个类别进行加权求和或者取最大，可以度量特征项 T_i 对于整个分类任务的信息量：

$$\chi^2_{\max}(T_i) = \max_{j=1,2,\cdots,M} \left\{\chi^2(T_i, C_j)\right\} \tag{5.20}$$

$$\chi^2_{\text{avg}}(T_i) = \sum_{j=1}^{M} p(c_j) \chi^2(T_i, C_j) \tag{5.21}$$

表 5.6 给出了用 χ^2 法对文本分类数据集（表 5.1）进行特征选择的结果。

表 5.6 用 χ^2 法对文本分类数据集（表 5.1）进行特征选择的结果

特　　征	χ^2
计算机 排球 运动会	3.9999
1958 2008 奥林匹克 博弈 场地 创 创建 第四 第五 东亚 高校 奖牌 锦标赛 军团 男女	1.3333
设立 双双 体育馆 新高 学子 于 预赛 在 之一 中 专业 总数 最早 北京 大学 理工	
的 夺冠 届 年 是 中国	0.0000

5.3.4 其他方法

文献（Nigam, 2000）提出了一种加权对数似然概率（weighted log-likelihood ratio, WLLR）指标用于度量特征项 t_i 和类别 c_j 的相关性：

$$\begin{aligned}\mathrm{WLLR}(t_i, c_j) &= p\left(t_i|c_j\right) \log \frac{p\left(t_i|c_j\right)}{p\left(t_i|\bar{c}_j\right)} \\ &= \frac{N_{t_i,c_j}}{N_{t_i,c_j} + N_{\bar{t}_i,c_j}} \log \frac{N_{t_i,c_j}(N_{t_i,\bar{c}_j} + N_{\bar{t}_i,\bar{c}_j})}{N_{t_i,\bar{c}_j}(N_{t_i,c_j} + N_{\bar{t}_i,c_j})}\end{aligned} \tag{5.22}$$

文献（Li et al., 2009a）进一步分析了 MI，IG，χ^2 和 WLLR 等六种特征选择方法，发现频率 $p\left(t_i|c_j\right)$ 和比率 $\dfrac{p\left(t_i|c_j\right)}{p\left(t_i|\bar{c}_j\right)}$ 是各种特征选择的两个基本度量，上述特征选择方法均可写成以上两个度量的组合形式。据此（Li et al., 2009a）提出了一种通用的加权频率和比率（weighted frequency and odd, WFO）方法：

$$\begin{aligned}\mathrm{WFO}\left(t_i, c_j\right) &= p\left(t_i|c_j\right)^{\lambda} \left(\log \frac{p\left(t_i|c_j\right)}{p\left(t_i|\bar{c}_j\right)}\right)^{1-\lambda} \\ &= \left(\frac{N_{t_i,c_j}}{N_{t_i,c_j} + N_{\bar{t}_i,c_j}}\right)^{\lambda} \left(\log \frac{N_{t_i,c_j}(N_{t_i,\bar{c}_j} + N_{\bar{t}_i,\bar{c}_j})}{N_{t_i,\bar{c}_j}(N_{t_i,c_j} + N_{\bar{t}_i,c_j})}\right)^{1-\lambda}\end{aligned} \tag{5.23}$$

假设特征选择得到以下降维后的向量空间：[计算机 排球 运动会 高校 大学]，利用降维后的向量空间对文本分类数据集（表 5.1）进行文本表示，得到表 5.7 所示的结果。

表 5.7 降维后的文本分类数据集

序号	原始文档	降维后的文档	类别
$train\_d_1$	北京 理工 大学 计算机 专业 创建 于 1958 年 是 中国 最早 设立 计算机 专业 的 高校 之一	大学 计算机 计算机 高校	教育
$train\_d_2$	北京 理工 大学 学子 在 第四 届 中国 计算机 博弈 锦标赛 中 夺冠	大学 计算机	教育
$train\_d_3$	北京 理工 大学 体育馆 是 2008 年 中国 北京 奥林匹克 运动会 的 排球 预赛 场地	大学 运动会 排球	体育
$train\_d_4$	第五届 东亚 运动会 中国 军团 奖牌 总数 创 新高 男女 排球 双双 夺冠	运动会 排球	体育
$test\_d_1$	北京 理工 大学 是 理工 为主 工理文 协调 发展 的 全国 重点 大学	大学 大学	
$test\_d_2$	复旦 大学 排球 队 获得 本届 大学生 运动会 排球 比赛 冠军	大学 排球 运动会 排球	

5.4 传统分类算法

一个文本经过文本表示和特征选择之后，就可以基于传统的机器学习算法进行文本分类。早期的文本分类模型包括 Rocchio、K-近邻分类器（K-nearest neighbor classifier）、决策树等，其后，得到了广泛使用的文本分类算法包括朴素贝叶斯模型、logistic 回归模型、最大熵模型、支持向量机和人工神经网络等。

5.4.1 朴素贝叶斯模型

贝叶斯模型属于生成式模型，它对样本的观测和类别状态的联合分布 $p(\boldsymbol{x}, y)$ 进行建模。在实际应用中，联合分布转换为类别的先验分布 $p(y)$ 与类条件分布 $p(\boldsymbol{x}|y)$ 乘积的形式：$p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x}|y)$。前者可以分别使用伯努利分布和类别分布建模两类和多类分类的类别先验概率，但类条件分布 $p(\boldsymbol{x}|y)$ 的估计问题是贝叶斯模型的难题。

在文本分类任务中，为了解决上述难题，需要对文本的类条件分布做进一步简化。一种通常的做法是忽略文本中的词序关系，假设各个特征词的位置是可以互换的，即我们前面所说的词袋模型。在数学上，这样的简化可以表示为在给定类别的条件下，词与词相互独立的假设。基于这一假设，类条件下的文本分布可以用多项分布刻画。这与判别式模型（discriminative model）中文本表示采用词频权重的向量空间模型的做法是一致的。基于以上条件的贝叶斯模型称为朴素贝叶斯模型（naïve Bayes, NB），它的本质是用混合的多项式分布刻画文本分布。虽然朴素贝叶斯模型具有很强的假设条件，但是在文本分类和情感分类任务中，仍然不失为简单高效的经典分类算法。

朴素贝叶斯模型是一种简化的贝叶斯分类器，对观测向量 $\boldsymbol{x}$ 和类别 y 的联合分布

$$p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x}|y) \tag{5.24}$$

进行建模。通常假设类别变量 y 服从伯努利分布（两类问题）或分类分布（categorical distribution）（多类问题），并根据实际任务对 $p(\boldsymbol{x}|y)$ 进行合理假设。朴素贝叶斯分类器之所以称作“朴素”，是因为它有一个很强的条件独立性假设：在给定类别的条件下，各个特征项之间相互独立。在图像分类等任务中，常常假设 $p(\boldsymbol{x}|y)$ 符合高斯分布，而在文本分类任务中，$p(\boldsymbol{x}|y)$ 常见的分布假设有两种（McCallum and Nigam, 1998）：多项分布模型（multinomial model）和多变量伯努利分布模型（multi-variate Bernoulli model）。其中多变量伯努利分布假设只关心特征项是否出现，而不记录出现的频次，在实际应用中，其效果往往不及多项分布假设。因此，在文本分类任务中，不加特别说明的朴素贝叶斯模型往往都是指基于多项式分布假设的朴素贝叶斯模型。

下面以多项分布模型为例介绍朴素贝叶斯模型。首先将一个文档 $\boldsymbol{x}$ 表示为一个词的序列

$$\boldsymbol{x} = \left[w_1, w_2, \cdots, w_{|\boldsymbol{x}|}\right] \tag{5.25}$$

在条件独立性假设下，$p(\boldsymbol{x}|y)$ 可以具有多项分布的形式：

$$\begin{aligned} p(\boldsymbol{x}|c_j) &= p([w_1, w_2, \cdots, w_{|\boldsymbol{x}|}]|c_j) \\ &= \prod_{i=1}^{V} p(t_i|c_j)^{N(t_i, \boldsymbol{x})} \end{aligned} \tag{5.26}$$

其中，V 是词汇表维度，t_i 表示词汇表中的第 i 个特征项。令 $\theta_{i|j}=p(t_i|c_j)$ 表示在 c_j 类条件下 t_i 出现的概率，$N(t_i,\boldsymbol{x})$ 表示在文档 $\boldsymbol{x}$ 中 t_i 的词频。

同时，我们以多类问题为例，假设类别 y 服从类别分布

$$p\left(y=c_j\right)=\pi_j \tag{5.27}$$

根据多项分布模型假设，$p\left(\boldsymbol{x},y\right)$ 的联合分布写为

$$p(\boldsymbol{x},y=c_j)=p(c_j)p(\boldsymbol{x}|c_j)=\pi_j\prod_{i=1}^{V}\theta_{i|j}^{N(t_i,\boldsymbol{x})} \tag{5.28}$$

其中，$\boldsymbol{\pi}$ 和 $\boldsymbol{\theta}$ 为模型参数。

朴素贝叶斯模型基于最大似然估计算法进行参数学习，给定训练集 $\{\boldsymbol{x}_k,y_k\}_{k=1}^{N}$，模型以对数似然函数 $L\left(\boldsymbol{\pi},\boldsymbol{\theta}\right)=\log\prod_{k=1}^{N}p\left(\boldsymbol{x}_k,y_k\right)$ 作为优化目标。对优化目标求导置零，求解得到模型的参数估计值：

$$\pi_j=\frac{\sum_{k=1}^{N}I(y_k=c_j)}{\sum_{k=1}^{N}\sum_{j'=1}^{C}I(y_k=c_{j'})}=\frac{N_j}{N} \tag{5.29}$$

$$\theta_{i|j}=\frac{\sum_{k=1}^{N}I(y_k=c_j)N(t_i,\boldsymbol{x}_k)}{\sum_{k=1}^{N}I(y_k=c_j)\sum_{i'=1}^{V}N(t_{i'},\boldsymbol{x}_k)} \tag{5.30}$$

从参数估计结果可以看出，在多项分布假设下，频率正是概率的最大似然估计值。例如，类别概率 π_j 的最大似然估计结果是训练集中第 j 类样本出现的频率；类条件下特征项概率的最大似然估计结果是第 j 类文档中所有特征项中 t_i 出现的频率。为了防止零概率情况的出现，常常对 $\theta_{i|j}$ 进行拉普拉斯平滑：

$$\theta_{i|j}=\frac{\sum_{k=1}^{N}I(y_k=c_j)N(t_i,\boldsymbol{x}_k)+1}{\sum_{i'=1}^{V}\sum_{k=1}^{N}I(y_k=c_j)N(t_{i'},\boldsymbol{x}_k)+V} \tag{5.31}$$

利用多项式朴素贝叶斯模型，在降维后的文本分类训练集（表 5.7）上进行模型学习，分别令 t_1= 计算机，t_2= 排球，t_3= 运动会，t_4= 高校，t_5= 大学，$y=1$ 表示教育类，$y=0$ 表示体育类，可以得到如表 5.8 所示的参数估计结果。

表 5.8 朴素贝叶斯多项式模型在降维后的文本分类训练集（表 5.7）上的训练结果

π_j	$p(y=1)=0.5$	$p(y=0)=0.5$
$\theta_{i\|j}$	$p(t_1\|y=1)=4/11$	$p(t_1\|y=0)=1/10$
	$p(t_2\|y=1)=1/11$	$p(t_2\|y=0)=3/10$
	$p(t_3\|y=1)=1/11$	$p(t_3\|y=0)=3/10$
	$p(t_4\|y=1)=2/11$	$p(t_4\|y=0)=1/10$
	$p(t_5\|y=1)=3/11$	$p(t_5\|y=0)=2/10$

基于上述模型，现对表 5.7 中的测试文档进行分类。令测试文档 $test\_d_1$ 的文本表示为 $\boldsymbol{x}_1$，它与教育类和体育类的联合概率分别为

$$p(\boldsymbol{x}_1, y=1) = p(y=1)\,p(t_5|y=1)^2 = 0.037$$
$$p(\boldsymbol{x}_1, y=0) = p(y=0)\,p(t_5|y=0)^2 = 0.020$$

进一步，根据贝叶斯公式计算可得属于两类的后验概率分别为

$$p(y=1|\boldsymbol{x}_1) = 0.649$$
$$p(y=0|\boldsymbol{x}_1) = 0.351$$

因此预测 $test\_d_1$ 属于教育类。

同理，测试文档 $test\_d_2$ 与两个类别的联合概率分别为

$$p(\boldsymbol{x}_2, y=1) = p(y=1)\,p(t_2|y=1)^2\,p(t_3|y=1)\,p(t_5|y=1) = 0.0001$$
$$p(\boldsymbol{x}_2, y=0) = p(y=0)\,p(t_2|y=0)^2\,p(t_3|y=0)\,p(t_5|y=0) = 0.0027$$

后验概率为

$$p(y=1|\boldsymbol{x}_2) = 0.036$$
$$p(y=0|\boldsymbol{x}_2) = 0.964$$

因此预测 $test\_d_2$ 属于体育类。

5.4.2 logistic 回归、softmax 回归与最大熵模型

虽然术语 logistic 回归中包含“回归”一词，但它却是一个地地道道的分类模型，它是一个线性二分类模型，它所决定的分类面是一个关于特征空间的超平面。以下仍从模型假设、学习准则和参数估计方法三个方面介绍 logistic 回归模型。

首先引入 sigmoid 函数 $\sigma(z) = \dfrac{1}{1+\mathrm{e}^{-z}}$。该函数可以将实数域映射为 $[0,1]$ 范围，因此常常作为概率描述。它的一阶导数具有以下优良性质：

$$\frac{\mathrm{d}\sigma(z)}{\mathrm{d}z} = \sigma(z)\,(1-\sigma(z)) \tag{5.32}$$

对于一个二分类问题，类别标记为 $y \in \{0,1\}$，特征向量为 $\boldsymbol{x}$，权重向量记作 $\boldsymbol{\theta}$。logistic 回归定义了给定 $\boldsymbol{x}$，$y \in \{0,1\}$ 的后验概率，形式如下：

$$\begin{cases} p(y=1|\boldsymbol{x};\boldsymbol{\theta}) = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}) \\ p(y=0|\boldsymbol{x};\boldsymbol{\theta}) = 1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}) \end{cases} \tag{5.33}$$

其中，特征向量的线性加权 $\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}$ 经过 sigmoid 函数映射为 $[0,1]$ 概率区间。上述两式可以写成如下简洁的形式：

$$\begin{aligned} p(y|\boldsymbol{x};\boldsymbol{\theta}) &= (h_{\boldsymbol{\theta}}(\boldsymbol{x}))^{y}(1-h_{\boldsymbol{\theta}}(\boldsymbol{x}))^{1-y} \\ &= \left(\frac{1}{1+\mathrm{e}^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}}\right)^{y}\left(1-\frac{1}{1+\mathrm{e}^{-\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{x}}}\right)^{1-y} \end{aligned} \tag{5.34}$$

对于式 (5.34) 给定的模型假设，logistic 回归基于最大似然估计准则进行参数学习。给定训练集 $\{(\boldsymbol{x}_i, y_i)\}, i = 1, 2, \cdots, N$，模型的对数似然函数为

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{n} y_i \log h_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + (1-y_i)\log\left(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right) \tag{5.35}$$

通常使用梯度上升法、随机梯度上升法求解上述对数似然函数的最优化问题，除此之外，BFGS（Broyden-Fletcher-Goldfarb-Shanno）算法、L-BFGS（limited-memory BFGS）等拟牛顿法算法在大规模数据的 logistic 回归模型中也使用广泛。

将 logistic 回归从两类分类问题推广到多类问题，称为多类 logistic 回归，也称为 softmax 回归（softmax regression）。softmax 回归常常作为深度神经网络的最后一层执行分类任务。

假设给定 $\boldsymbol{x}$，类别 $y = c_j$ 的后验概率具有以下 softmax 函数形式：

$$\begin{aligned} p(y=c_j|\boldsymbol{x};\boldsymbol{\Theta}) &= h_j(\boldsymbol{x}) \\ &= \frac{\exp(\boldsymbol{\theta}_j^{\mathrm{T}}\boldsymbol{x})}{\sum_{l=1}^{C}\exp(\boldsymbol{\theta}_l^{\mathrm{T}}\boldsymbol{x})}, \quad j = 1, 2, \cdots, C \end{aligned} \tag{5.36}$$

其中，参数空间 $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_j\}, j = 1, 2, \cdots, C$。

根据模型假设，给定训练集 $\{(x_i, y_i)\}$，softmax 回归的对数似然函数为

$$L(\boldsymbol{\Theta}) = \sum_{i=1}^{N}\sum_{j=1}^{C} I(y_i = c_j)\log\, h_j(\boldsymbol{x}_i) \tag{5.37}$$

从信息论的角度，softmax 回归模型的负对数似然函数可以看作样本类别的真实分布与预测分布的交叉熵，因此也常被称为交叉熵损失。

值得一提的是，softmax 回归和朴素贝叶斯可以看作一个“判别式-生成式”模型对，这在文献（Ng and Jordan, 2002）中有具体的论述。

在自然语言处理领域，还有一个与之殊途同归、引入原理不同但形式非常相似的模型，称为最大熵模型。该模型假设给定状态条件下观测值的后验概率分布具有对数线性方程的形式，并利用最大似然估计或最大熵准则进行参数训练。

需要注意的是，最大熵模型中的特征与 softmax 回归中的特征定义略有区别。softmax 回归是在向量空间模型中定义特征向量，支持连续的实数特征，而最大熵模型是利用特征函数描述样本观测与类别的关联性，只支持 0-1 特征。特征函数（feature function）描述输入 $\boldsymbol{x}$ 和输出 y 之间已知的事实关系：

$$f_i(\boldsymbol{x}, y) = \begin{cases} 1, & \boldsymbol{x} \text{满足某一事实，且} y \text{为某一类别} \\ 0, & \text{其他} \end{cases} \tag{5.38}$$

以表 5.7 的文本分类数据集为例，最大熵模型的特征可以构造为：输入 $\boldsymbol{x}$ 包含“大学”且输出 y 为“教育”类；输入 $\boldsymbol{x}$ 包含“运动会”且输出 y 为“体育”类；输入 $\boldsymbol{x}$ 第一个词为“大学”、第二个词为“运动会”且输出 y 为“体育”类等。当最大熵模型的特征模板与 softmax 回归的向量空间模型定义一致时，最大熵模型和 softmax 回归模型是等价的。

5.4.3 支持向量机

支持向量机 (support vector machine，SVM)是统计机器学习领域富有盛名的分类算法。它的两个核心思想是：①寻找具有最大类间距离的决策面；②通过核函数（kernel function）在低维空间计算并构建分类面，将低维不可分问题转化为高维可分问题。SVM 具有深厚的统计学习理论背景，它基于结构风险最小化理论在特征空间中构建最优分类超平面，使学习器得到了全局最优化，并且在整个样本空间的期望风险以某个概率满足一定的上界约束。基于线性核函数的支持向量机在文本分类中有着非常广泛的应用。

上文提及的 logistic 回归模型都是线性分类模型。对于一个线性可分的两分类任务，如何找到最优的线性分类面，不同的分类器具有不同的训练准则。如感知机依据感知机准则，逻辑回归模型依据最小交叉熵准则等。线性 SVM 也是一种两分类任务的线性分类模型，它所采用的分类准则称为最大间隔准则（maximum margin criterion）。

对于线性分类模型

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \tag{5.39}$$

其线性分类面为 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b = 0$。SVM 采用最大分类间隔（maximum margin）作为模型训练准则。最大分类间隔准则用公式可以表示为

$$\begin{aligned} &\max_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t.} \quad y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b\right) \geqslant 1, \quad i = 1, 2, \cdots, N \end{aligned} \tag{5.40}$$

可以看出，这是一个标准的二次优化问题，其目标函数是二次的，约束条件是线性的。该问题可以用任何现成的二次规划（quadratic programming）优化包进行求解。

鉴于上述优化问题的特殊结构，SVM 通过拉格朗日对偶法将式 (5.40) 所示的原问题转化为下列对偶问题以进行更加高效的求解：

$$\begin{aligned}\max_{\boldsymbol{\alpha}} \ & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \\ \text{s.t.} \quad & \alpha_i \geqslant 0, i = 1, 2, \cdots, N \\ & \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{5.41}$$

其中，$\alpha_i \geqslant 0$ 是拉格朗日乘子（Lagrange multiplier）。对偶问题符合 KKT 条件（KKT（Karush-Kuhn-Tucker）condition），根据 KKT 条件：仅在分类边界上的样本 $\alpha_i > 0$，其余样本 $\alpha_i = 0$，并由此可得分类面仅由分类边界上的样本支撑。这也是支持向量机得名的由来。

在实际应用中，为了排除训练集中的野点对分类面的影响，通常定义软间隔准则，对最大分类间隔准则进行如下修正：

$$\begin{aligned}\max_{\boldsymbol{w},b} \ & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b\right) \geqslant 1 - \xi_i \\ & \xi_i \geqslant 0, i = 1, 2, \cdots, N \end{aligned} \tag{5.42}$$

其中，ξ_i 为容错因子，C 为容错项的权重参数。其相应的对偶问题为

$$\begin{aligned}\max_{\boldsymbol{\alpha}} \ & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \\ \text{s.t.} \quad & 0 \leqslant \alpha_i \leqslant C, i = 1, 2, \cdots, N \\ & \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{5.43}$$

同时，SVM 引入核函数理论将低维的线性不可分问题转化为高维的线性可分问题。核函数（kernel function）定义为核数据在高维空间的内积：

$$K(\boldsymbol{x}, \boldsymbol{z}) = \varphi(\boldsymbol{x})^{\mathrm{T}}\varphi(\boldsymbol{z}) \tag{5.44}$$

根据式 (5.43)，SVM 中样本 $\boldsymbol{x}$ 所涉及的运算均为内积运算。因此，无须知道低维到高维映射的具体形式，只需知道核函数的形式，就可以在高维空间建立线性 SVM 模型。此时对应的对偶问题为

$$\begin{aligned}\max_{\boldsymbol{\alpha}} \ & W(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ \text{s.t.} \quad & 0 \leqslant \alpha_i \leqslant C, i = 1, 2, \cdots, N \\ & \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{5.45}$$

决策函数为

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^* y_i \langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}) \rangle + b^* = \sum_{i=1}^{N} \alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b^* \tag{5.46}$$

常见的核函数包括：

- 线性核函数：$K(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{z}$；
- 多项式核函数：$K(\boldsymbol{x}, \boldsymbol{z}) = \left(\boldsymbol{x}^{\mathrm{T}} \boldsymbol{z} + c\right)^d$；
- 径向基核函数：$K(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{z}|^2}{2\delta^2}\right)$。

另外还有 sigmoid 核函数、pyramid 核函数、string 核函数和 tree 核函数等。由于文本分类任务中特征空间维度较高，通常来说都是线性可分的，因此线性核函数是最常被选择的。

上文介绍了利用对偶优化将原问题转化为式（5.45）所示的对偶问题，下一步还需进一步求解对偶问题得到最优参数 $\boldsymbol{\alpha}^*$，比较有代表性的方法是 SMO（sequential minimal optimization）算法（Platt, 1998），这里不再多述。

SVM 作为一种传统的统计分类方法，在 20 世纪末和 21 世纪初的文本分类任务中有着非常广泛的应用。根据论文（Yang and Liu, 1999）的实验结果，SVM 对主题文本分类的效果明显好于 NB、多层前馈神经网络和分段线性拟合等方法，与 k-近邻方法效果相当，甚至更好。在（Pang et al., 2002）给出的实验中，利用一元语法特征，SVM 在电影评论语料上的分类性能高于 NB 和 ME 方法。

5.4.4 集成学习

集成学习（ensemble learning）也称组合分类器，就是将多个分类器（弱分类器）的输出融合为一个精度更高的分类器（强分类器）的过程，近年来集成学习成为机器学习领域研究的一个重要分支。产生多个基分类器的方式主要有三种：①通过训练不同的数据集产生不同的分类器；②基于不同的特征集合进行训练得到不同的基分类器；③通过不同的分类算法产生不同的基分类器。

著名的 Bagging（bootstrap aggregating）算法和 Boosting 算法均以第一种方式产生基分类器。Bagging 算法是（Breiman, 1996）提出的，其思想是对训练集按可放回的方式抽取训练样本，为每个基分类器构造出一个跟原始训练集规模一致的训练集，从而训练出不同的基分类器。Boosting 算法是另一类代表性的集成学习算法，它首先给每个样本赋予相同的权重，然后训练第一个基分类器，并用它对训练集进行测试，对那些分类错误的样本提高权重，最后用调整的加权训练集训练第二个基分类器，如此重复，直至学习到一个足够好的分类器。Boosting 算法有许多不同的变种，（Freund and Schapire,

1996）提出的 AdaBoost 算法是其中的代表。分类器集成的算法也非常多，常见的有三类：固定的规则、加权规则和元学习方法。

集成学习在文本分类领域获得了成功的应用。（Larkey and Croft, 1996）是早期具有代表性的一项工作，它将不同的机器学习算法进行组合，得到了一个比基分类器性能更优的组合分类器。（Schapire and Singer, 2000）提出了一种基于 Boosting 算法的文档分类系统 BoosTexter，在当时表现出了比传统算法更好的性能。

5.5 深度神经网络方法

传统的文本表示和分类算法依赖人工设计的特征工程，具有维度高、稀疏性强、表达能力差、不能自动学习等诸多缺点。近年来，以深度神经网络为代表的深度学习技术自从在语音识别和图像处理领域取得了较大突破之后，以其强大的特征自学习能力（尤其是端到端的联合学习框架），在自然语言处理领域获得了广泛的应用，在包括文本分类在内的诸多任务上都取得了较大的进展，目前已经发展成为主流方法。

以下简要介绍几种用于文本分类的神经网络方法。

5.5.1 基于前馈神经网络的文本分类方法

多层感知器（multi-layer perceptron, MLP）是一种前向结构的人工神经网络，它通过全连接的方式映射一组输入向量到一组输出向量，若干神经元被分层组织在一起便组成了神经网络。与单个神经元相比，多层感知器增加了隐藏层 (hidden layer)，并在隐藏层的神经元中增加了激活函数，用于进行非线性变换，从而使多层感知器能够表示所有的函数映射。

尽管深度神经网络近十年才在包括文本分类在内的自然语言处理领域得到大规模应用，但 20 世纪末已经出现了以多层前馈神经网络为代表的文本分类神经网络方法（Yang and Liu, 1999）。但是，那时的神经网络还只是被当作一个分类器模块，在传统的文本分类系统框架下，文本通过向量空间模型被表示为一个稀疏向量之后作为神经网络的输入层，整个模型并没有特征自学习的能力。同时，由于当时数据量较小，以人工神经网络为代表的非线性分类器并没有取得显著优越的性能，加之运算开销较大，因此并没有得到青睐。

近年来，随着数据量的增大、运算性能的提高和从特征表示到分类，以及端到端一体化学习框架的应用，以深度学习重新冠名的人工神经网络模型，包括卷积神经网络（convolutional neural network，CNN）、循环神经网络（recurrent neural network）、长短时记忆（long-short term memory，LSTM）网络等，在文本分类任务上取得了巨大的成功。

5.5.2 基于卷积神经网络的文本分类方法

正如 3.5.2 节所述，卷积神经网络 (convolutional neural network，CNN)是一种前馈

神经网络，它由一个或多个卷积层（convolution layer）与池化层（pooling layer）的连接以及最后的全连接层（fully connected layer）构成。与多层前馈神经网络相比，卷积神经网络在结构上具有局部连接、权重共享和空间次采样的特点，具有较少的网络参数。图 5.4 给出了一个基于 CNN 的文本分类模型基本结构，它由输入层、卷积层、池化层、全连接层和输出层组成。

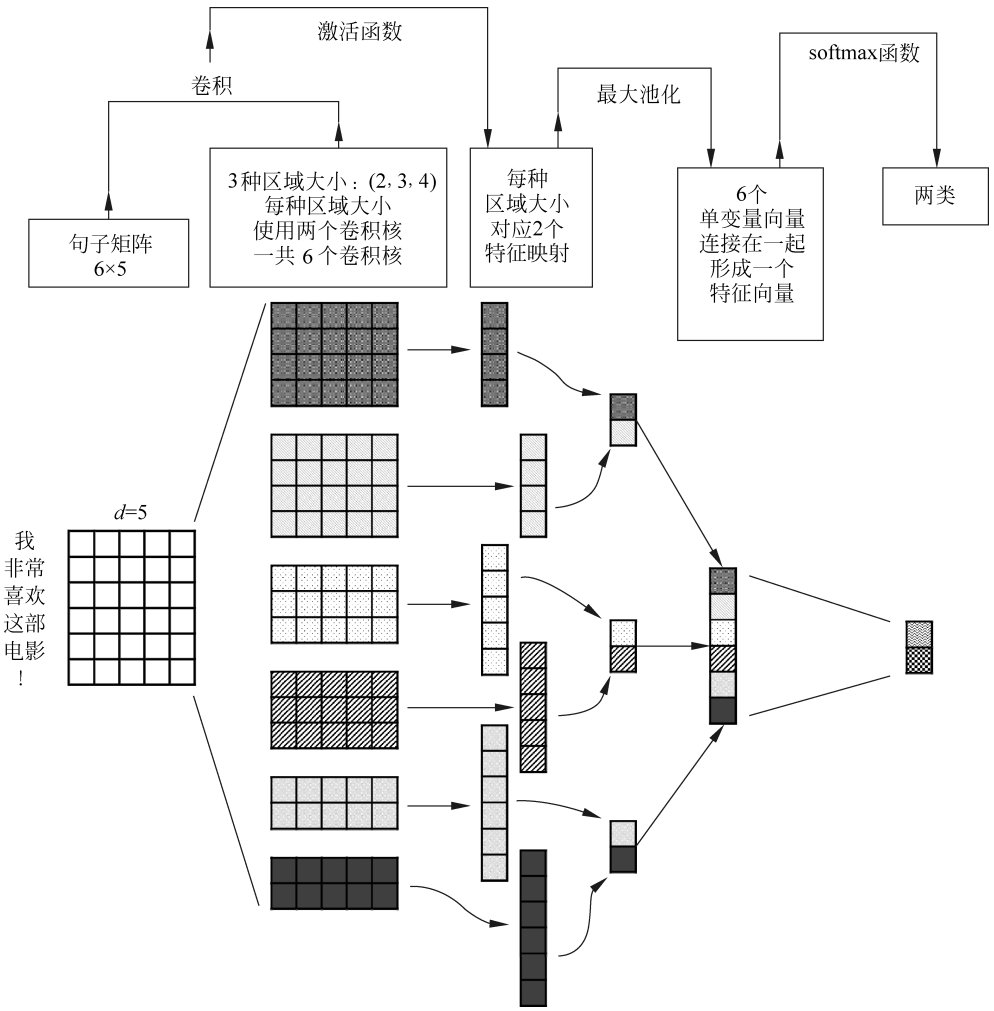


图 5.4 基于 CNN 的文本分类模型结构

基于 CNN 进行文本分类通常需要如下几个步骤：

（1）对输入文本进行形态处理（汉语分词）等预处理后得到词序列，使用词向量对词进行初始化，得到输入文本的矩阵表示形式，作为神经网络的输入。

（2）通过卷积层对输入进行特征提取。以图 5.4 为例，卷积层设置了 2×5、3×5、4×5 三种尺寸的卷积核（convolution kernel），每个尺寸具有两个卷积核。需要说明的是，在文本处理中对输入文本的表示矩阵进行卷积操作时，通常只在一个方向上进行二维卷积（即卷积核的宽度与词向量的维度保持一致），同时设置卷积操作的步长为 1，使

用每个卷积核对输入文本的表示矩阵进行卷积操作，每个卷积核对应得到一个输入文本的向量表示。

（3）池化层对卷积层输出的特征向量分别进行下采样，之后拼接得到进一步抽象的文本表示。不同长度的文本经过卷积层输出的特征向量具有不同的维度，池化层将这种特征向量转化为相同的维度。如图 5.4 所示，对每个特征向量进行最大池化，拼接后得到长度为卷积核数目的特征向量。通过全连接层将池化层获得的向量表示映射到样本的标注空间，维度与类别数一致，再通过 softmax 函数输出每一类的预测概率，最终完成文本分类。

基于 CNN 的句子文本分类方法（Kim, 2014）在主题分类、情感分类等任务上都取得了超越经典机器学习方法的效果。（Kalchbrenner et al., 2014）提出了一种动态卷积神经网络模型（dynamic convolutional neural network），在卷积层对句子的词向量矩阵进行二维卷积后，使用动态 k-max 池化操作对其进行下采样，且使用最重要的几个特征值表示局部特征。（Zhang et al., 2015）针对英文单词是由字符组成的这个特性，提出了字符级的卷积神经网络（character-level CNN），在更细粒度上对英文单词进行卷积处理，在相关数据集上取得了当时最佳的分类效果。

5.5.3 基于循环神经网络的文本分类方法

目前卷积神经网络主要应用在图像处理和机器视觉等领域，而自然语言处理的对象通常是一段具有循环结构的文本序列，因此更加适合利用循环神经网络（recurrent neural network, RNN）进行文本的表示和分类。

本书 3.3.1 节已经详细介绍了 LSTM 和 GRU 等 RNN 模型，因此本节不再描述其模型细节，重点介绍基于 RNN 的句子级和文档级文本分类方法。

1. 基于 RNN 的句子级文本分类模型

本节以句子级情感分类为例，介绍如何使用 RNN 完成文本分类任务。假设输入为“我非常喜欢这部电影”，其类别标签为正向情感。

如图 5.5 所示，首先利用预先训练好的词向量获得句子的初始表示 $[x_1, x_2, \cdots, x_T]$，按时序将各个词的词向量 $\boldsymbol{x}_t$ 输入 Bi-LSTM 中：

$$\overrightarrow{\boldsymbol{c}}_t, \overrightarrow{\boldsymbol{h}}_t = \text{LSTM}\left(\overrightarrow{\boldsymbol{c}}_{t-1}, \overrightarrow{\boldsymbol{h}}_{t-1}, \boldsymbol{w}_t\right) \tag{5.47}$$

$$\overleftarrow{\boldsymbol{c}}_t, \overleftarrow{\boldsymbol{h}}_t = \text{LSTM}\left(\overleftarrow{\boldsymbol{c}}_{t+1}, \overleftarrow{\boldsymbol{h}}_{t+1}, \boldsymbol{w}_t\right) \tag{5.48}$$

相应得到其隐层状态向量：

$$\boldsymbol{h}_t = \left[\overrightarrow{\boldsymbol{h}}_t, \overleftarrow{\boldsymbol{h}}_t\right] \tag{5.49}$$

将句子全部词语处理完毕，得到隐层状态矩阵 $[\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_T]$。

基于注意力机制计算每个词的权重 α_t：

$$\alpha_t = \text{softmax}\left(\boldsymbol{u}_t^{\mathrm{T}}\boldsymbol{q}\right) \tag{5.50}$$

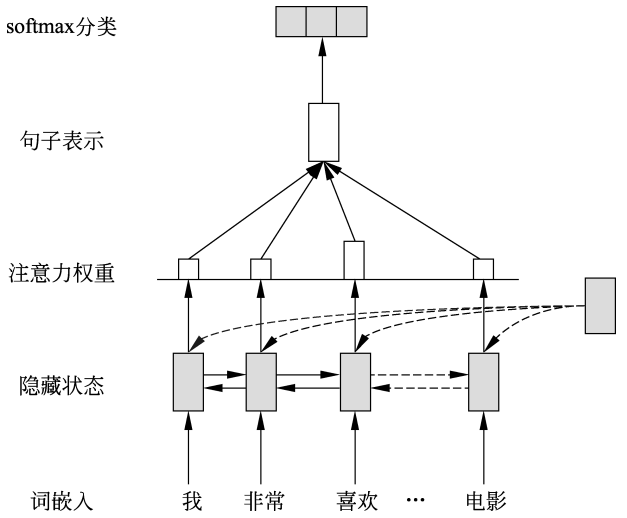


图 5.5 句子级文本分类示例

其中，$\boldsymbol{u}_t = \tanh(\boldsymbol{W}\boldsymbol{h}_t + \boldsymbol{b})$，$\boldsymbol{q}$ 为查询向量。基于权重对各个词的隐层状态进行线性加权，得到句子的最终表示向量：

$$\boldsymbol{r} = \sum_t \boldsymbol{\alpha}_t \boldsymbol{h}_t \tag{5.51}$$

将最后获得的句子表示向量后送入 softmax 层对文本进行分类，得到各类别的预测概率：

$$p = \text{softmax}(\boldsymbol{W}_c \boldsymbol{r} + \boldsymbol{b}_c) \tag{5.52}$$

其中，$\boldsymbol{W}_c$ 和 $\boldsymbol{b}_c$ 为权重矩阵和偏置项。

模型以句子真实标注 $\boldsymbol{y}$ 和分类预测 $\boldsymbol{p}$ 的交叉熵 E 作为优化目标：

$$E = -\sum_{j=1}^{C} y_j \log p_j \tag{5.53}$$

并利用 BPTT 算法进行模型的参数学习。

2. 层次化的文档级文本分类模型

文档级文本分类是指在整个文档粒度上的文本分类，每个文档包含一个类别标签。基于 RNN 进行文档级文本分类有一种简单做法，即将文档视为一个长句子，利用 RNN 对这个长句子进行编码并分类，但是这种做法没有考虑文档中的层次结构。

正如 3.5.2 节所述，由于文档通常包含多个句子，每个句子又包含多个词，因此可以按照“词-句子-文档”的层次结构来对文档表示和建模。（Tang et al., 2015）首先使用 LSTM（或 CNN）对句子中的词序列进行向量表示和编码，然后通过 Gated RNN 对文档中的句子序列进行编码，在文档级情感分类任务上获得了较好的性能。（Yang et al., 2016）进一步提出了一种层次注意力 RNN 模型，按照“词-句子-文档”的层次结构进行

文档级文本分类，结构如图 5.6 所示。该模型主要包含五个部分：词序列编码器、词级注意力层、句子序列编码器、句子级注意力层和 softmax 分类层。模型使用 GRU 作为基本神经网络单元。假设文档包含 L 个句子，每个句子包含 T_i 个词，s_i 表示第 i 个句子，x_{it} 表示第 i 个句子的第 t 个词的词向量。

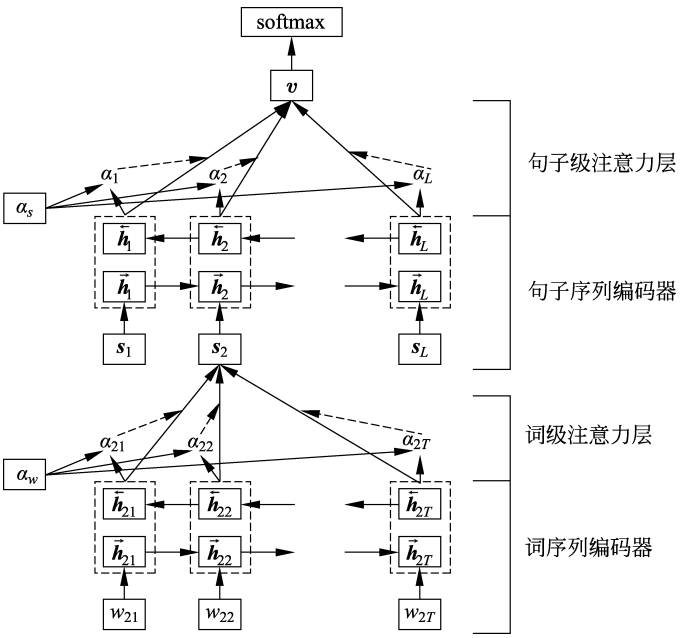


图 5.6 层次化的文档级文本分类模型

我们自下往上依次介绍这五个部分：

（1）词序列编码层。对于每个句子，词向量初始化后送入到 Bi-GRU 中，得到每个词的前向隐层状态向量 $\overrightarrow{\boldsymbol{h}}_{it}$ 和后向隐层状态向量 $\overleftarrow{\boldsymbol{h}}_{it}$，拼接后得到每个词的表示向量 $\boldsymbol{h}_{it} = \left[\overrightarrow{\boldsymbol{h}}_{it}, \overleftarrow{\boldsymbol{h}}_{it}\right]$。

（2）词级注意力层。针对每个词在句子表示中不同的重要性，计算每个词的权重 $\alpha_{it} = \dfrac{\exp\left(\boldsymbol{u}_{it}^{\mathrm{T}}\boldsymbol{u}_w\right)}{\sum\limits_t \exp\left(\boldsymbol{u}_{it}^{\mathrm{T}}\boldsymbol{u}_w\right)}$，其中 $\boldsymbol{u}_{it} = \tanh\left(\boldsymbol{W}_w\boldsymbol{h}_{it} + \boldsymbol{b}_w\right)$，$\boldsymbol{u}_w$ 是词级别的上下文向量，它可以看作查询语句“哪个词更重要”的表示，在模型中被随机地初始化并与模型其他参数一体化学习。最后将句子中各词的隐层表示线性加权，得到该句子的表示向量 $\boldsymbol{s}_i = \sum\limits_t \alpha_{it}\boldsymbol{h}_{it}$。

（3）句子序列编码层。经过词注意力层之后，每个句子得到了其表示向量。整个文档包含的句子组成了句子序列。与词序列编码层类似，把句子作为单元送入 Bi-GRU，得到该句子的前向 $\overrightarrow{\boldsymbol{h}}_i$ 和后向隐层表示向量 $\overleftarrow{\boldsymbol{h}}_i$，拼接得到句子最终的隐层表示向量 $\boldsymbol{h}_i = \left[\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i\right]$。

（4）句子级注意力层。为了区分不同句子对于文档表示的重要性，再次引入注意力机制，计算每个句子的权重 $\alpha_i = \dfrac{\exp\left(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{u}_s\right)}{\sum\limits_i \exp\left(\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{u}_s\right)}$，其中 $\boldsymbol{u}_i = \tanh\left(\boldsymbol{W}_s\boldsymbol{h}_i + \boldsymbol{b}_s\right)$。对句子的表示进行线性加权后得到文档的表示向量 $\boldsymbol{v} = \sum\limits_i \alpha_i \boldsymbol{h}_i$。

（5）softmax 分类层。将文档的表示向量 $\boldsymbol{v}$ 送到 softmax 层进行文档分类，计算 $\boldsymbol{p} = \mathrm{softmax}(\boldsymbol{W}_c\boldsymbol{v} + \boldsymbol{b}_c)$，其中，$\boldsymbol{W}_c$ 和 $\boldsymbol{b}_c$ 分别是权重矩阵和偏置项。模型以文档真实标注 $\boldsymbol{y}$ 和分类预测 $\boldsymbol{p}$ 的交叉熵作为优化目标，并基于 BPTT 算法进行模型参数学习。

5.6 文本分类性能评估

假设一个文本分类任务共有 M 个类别，类别名称分别为 $C_1, C_2, \cdots, C_M$。在完成分类任务以后，对于每一类都可以统计出真正例、真负例、假正例和假负例四种情形的样本数目。

- 真正例（true positive, TP）：模型正确预测为正例（即模型预测属于该类，真实标签属于该类）。
- 真负例（true negative, TN）：模型正确预测为负例（即模型预测不属于该类，真实标签不属于该类）。
- 假正例（false positive, FP）：模型错误预测为正例（即模型预测属于该类，真实标签不属于该类）。
- 假负例（false negative, FN）：模型错误预测为负例（即模型预测不属于该类，真实标签属于该类）。

对所有的类别统计出 TP，TN，FP 和 FN 之后，可以得到表 5.9 所示的微观统计值。

表 5.9 分类性能的微观统计值

类别	TP	FP	FN	TN
C_1	TP_1	FP_1	FN_1	TN_1
C_2	TP_2	FP_2	FN_2	TN_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
C_M	TP_M	FP_M	FN_M	TN_M

文本分类任务的性能评价指标通常包括如下几种。

1. 召回率、精确率和 F_1 值

假设 $j \in \{1, 2, \cdots, M\}$ 是类别序号，可以为每一类定义以下指标：

（1）召回率（recall）

$$R_j = \frac{\mathrm{TP}_j}{\mathrm{TP}_j + \mathrm{FN}_j} \tag{5.54}$$

（2）精确率（precision）

$$P_j = \frac{\mathrm{TP}_j}{\mathrm{TP}_j + \mathrm{FP}_j} \tag{5.55}$$

我们希望一个好的分类系统同时具有较高的召回率和精确率，但两者常常是矛盾的。单一地追求一种指标的提高，势必造成另一个指标的降低。因此，通常定义 F_1 值为精确率与召回率的调和平均数，以综合评价两个指标的共同作用。

（3）F_1 值

$$F_1 = \frac{2PR}{P + R} \tag{5.56}$$

在某些应用中，为了区分召回率和精确率的重要性，定义更为一般的 F_β 值：

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \tag{5.57}$$

当 $\beta = 1$ 时，F_β 退化为标准的 F_1 值。

2. 正确率、宏平均和微平均

召回率、精确率和 F_1 值只能评估某一类数据的分类性能。为了考察整个分类任务的性能，定义分类正确率为

$$\mathrm{Acc} = \frac{\#\mathrm{Correct}}{N} \tag{5.58}$$

其中，N 为样本总数，#Correct 为其中被模型正确预测的样本数。

除此之外，还可以使用各类指标的宏平均（macro-average)和微平均（micro-average）评估整个分类任务的性能。从名称上可以看出，宏平均值是先计算各类的宏观指标（召回率、精确率），再按类求平均，而微平均是将微观指标（TP，TN，FP 和 FN）按类求平均后，再计算召回率、精确率和 F_1 值。

宏平均的召回率、精确率和 F_1 值定义分别为

$$\mathrm{Macro\_P} = \frac{1}{C}\sum_{j=1}^{C}\frac{\mathrm{TP}_j}{\mathrm{TP}_j + \mathrm{FP}_j} \tag{5.59}$$

$$\mathrm{Macro\_R} = \frac{1}{C}\sum_{j=1}^{C}\frac{\mathrm{TP}_j}{\mathrm{TP}_j + \mathrm{FN}_j} \tag{5.60}$$

$$\mathrm{Macro\_F_1} = \frac{2 \times \mathrm{Macro\_P} \times \mathrm{Macro\_R}}{\mathrm{Macro\_P} + \mathrm{Macro\_R}} \tag{5.61}$$

微平均的召回率、精确率和 F_1 值定义分别为

$$\mathrm{Micro\_P} = \frac{\sum_{j=1}^{C}\mathrm{TP}_j}{\sum_{j=1}^{C}(\mathrm{TP}_j + \mathrm{FP}_j)} \tag{5.62}$$

$$\text{Micro\_R} = \frac{\sum_{j=1}^{C} \text{TP}_j}{\sum_{j=1}^{C} (\text{TP}_j + \text{FN}_j)} \tag{5.63}$$

$$\text{Micro\_F}_1 = \frac{2 \times \text{Micro\_P} \times \text{Micro\_R}}{\text{Micro\_P} + \text{Micro\_R}} \tag{5.64}$$

在二分类且类别互斥的情况下，Micro_R，Micro_P，Micro_F_1 都与正确率 Acc 相等。

对于表 5.10 给出的两分类问题的分类结果，可以计算得到上述所有指标，见表 5.11。

表 5.10 二分类的分类结果示例

预测/真实	正类 (+)	负类 (−)	全部
正类 (+)	250	20	270
负类 (−)	50	180	230
全部	300	200	500

表 5.11 针对表 5.10 分类结果的评估指标

	TP	FP	FN	TN	Recall	Precision	F_1	Acc
正类 (+)	250	20	50	180	0.8333	0.9259	0.8772	0.8600
负类 (−)	180	50	20	250	0.9000	0.7826	0.8372	
宏平均					0.8667	0.8543	0.8605	
微平均					0.8600	0.8600	0.8600	

3. PR 曲线、ROC 曲线

在分类问题中，模型进行样本类别预测本质上基于模型输出值与阈值的比较。例如，logistic 回归模型的阈值为 0.5，模型的输出值大于 0.5 时预测为正类，小于 0.5 时预测为负类。为了更加全面地评价分类器在不同召回率情况下的分类效果，可以通过调整分类器的阈值，将按输出排序的样本序列分割为两部分，大于阈值的预测为正类，小于阈值的预测为负类，从而得到不同的召回率和精确率。如设置阈值为 0 时，召回率为 1；设置阈值为 1 时，则召回率为 0。以召回率作为横轴、精确率作为纵轴，可以绘制出精确率-召回率（precision-recall, PR）曲线。理论上讲，PR 曲线越靠近右上方越好，如果一个模型的 PR 曲线在右上方“包住”另一模型的 PR 曲线，则说明其分类性能明显优于后者。对于 PR 曲线相交的情形，可以通过计算 PR 曲线下方的面积度量分类的性能，面积越大，分类性能越好。更简单地，11 点平均精确率法通过调整分类器，使其召回率分别为 0，0.1，0.2，0.3，0.4，0.5，0.6，0.7，0.8，0.9，1.0，然后利用这 11 点的平均精确率衡量分类器的性能。

类似地，以假正率（false positive rate）作为横坐标，以真正率（true positive rate）（即召回率）作为纵坐标，绘制出的曲线称为 ROC（receiver operating characteristic）曲线。ROC 曲线下的面积称为 AUC（area under ROC curve），AUC 曲线越靠近左上方越好。AUC 值越大，说明分类器性能越好。

5.7 进一步阅读

一方面，基于统计机器学习的分类模型大体可以分成两类：生成式模型（generative model）和判别式模型（discriminative model）。若在分类模型中把样本特征向量 $\boldsymbol{x}$ 作为观测值，把样本类别 y 作为状态值，判别式模型认为 y 由 $\boldsymbol{x}$ 决定，直接对给定观测值 $\boldsymbol{x}$ 条件下状态 y 的后验概率 $p(y|\boldsymbol{x})$ 或者两者的映射关系 $y = f(\boldsymbol{x})$ 进行建模，它从 $\boldsymbol{x}$ 中提取特征，学习模型参数，使得后验概率符合一定形式的最优。生成式模型则就每个状态 y 按照分布 $p(\boldsymbol{x}|y)$ 生成观测值 $\boldsymbol{x}$，对观测值和状态值的联合分布 $p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x}|y)$ 建模，并且通过最大似然估计来学习模型参数。在文本分类领域，常见的判别式分类模型包括 logistic 回归、最大熵模型、支持向量机和人工神经网络等，典型的生成式分类模型则包括朴素贝叶斯模型等。

另一方面，本章介绍的模型都是针对文本整体信息的分类，并未涉及针对文本序列结构信息的预测。考虑一个由多个节点组成的文本序列 $\boldsymbol{x}$，文本分类任务中 $\boldsymbol{x}$ 对应一个状态标签 y；如果 $\boldsymbol{x}$ 中的每个节点 $\boldsymbol{x}_t$ 都对应一个状态标签 y_t，则文本分类任务就转化为了文本序列标注任务。序列标注任务的本质是对序列中的每个节点进行分类，并且在分类预测中考虑序列中节点间的关系，以寻求在序列信号上的全局最优。序列标注中的常见模型包括隐马尔可夫模型（hidden Markov model, HMM）、条件随机场（conditional random field, CRF）等。HMM 可以看作朴素贝叶斯模型从分类问题向序列标注问题的扩展，HMM 除了以发射概率来建立 $\boldsymbol{x}_t$ 和 y_t 的关系，还利用状态转移概率来建立 y_{t-1} 和 y_t 的关系，从而实现序列的关系学习。类似地，CRF 模型是最大熵模型从分类问题向序列标注问题的扩展，CRF 借鉴了最大熵模型的对数线性模型（log-linear model）假设，定义了相似的观测状态特征函数，此外 CRF 还定义了一个状态转移特征函数用于学习序列中的结构关系。更多关于 HMM 和 CRF 等序列标注模型的介绍可参考（宗成庆，2013）。

循环神经网络天然具备同时处理文本分类和序列标注问题的能力，如果 RNN 编码序列的每个节点都进行分类预测，则形成了序列标注问题；如果将每个节点的输出通过语义组合形成文档级别的输出再进行分类，则形成文档分类任务。这种高度的灵活性也是循环神经网络针对文本建模的一大优势。

习　　题

5.1　除了多项分布，朴素贝叶斯模型还可以基于多变量伯努利分布来刻画类条件分布 $p(\boldsymbol{x} \,|\, y)$，请阅读（McCallum and Nigam, 1998），并阐述这两种假设之间的区别。

5.2 基于多变量伯努利分布假设，在表 5.7 给出的降维后的文本分类数据集上，计算朴素贝叶斯模型在训练集上的参数学习结果和在测试集上的分类预测结果。

5.3 从模型假设和参数学习两个角度，阐述 softmax 回归模型和最大熵模型之间的异同。

5.4 线性核函数 SVM 和 RBF 核函数 SVM，哪一个更适用于文本分类任务？给出你的理由。

5.5 多层前馈神经网络与卷积神经网络的主要区别是什么？对于文本分类，卷积神经网络为何可以捕获 n 元语法特征？

5.6 循环神经网络和卷积神经网络有什么本质区别？哪个更适用于文本分类任务？为什么？