吉林工程技术师范学院
JILIN ENGINEERING NORMAL UNIVERSITY

吉林工程技术师范学院

毕业论文

基于机器学习的中文情感识别研究

吉林工程技术师范学院 吴泞宇

吴泞宇

2023 年 5 月

编号 SZSJKX1947084132

吉林工程技术师范学院
JILIN ENGINEERING NORMAL UNIVERSITY

基于机器学习的中文情感识别研究

Research on Chinese Emotion Recognition based on Machine Learning

学 生 姓 名 吴泞宇

专 业 数据科学与大数据技术

学 号 1947084132

指 导 教 师 田丽华

职 称 教授

学 院 数据科学与人工智能学院

二〇二三年五月

毕业论文原创承诺书

1．本人承诺：所呈交的毕业论文《基于机器学习的中文情感识别研究》，符合《吉林工程技术师范学院毕业设计（论文）工作管理规定》中的各项要求，是在教师的指导下独立完成，无弄虚作假，抄袭剽窃情况。

2．本人在毕业论文中引用他人的观点和研究成果，均在文中加以注释或以参考文献形式列出，对本文的研究工作做出重要贡献的个人和集体均已在文中注明。

3．在毕业论文中出现的任何侵犯他人知识产权的行为，由本人承担相应的法律责任。

4．本人知道学校关于保存、使用毕业论文的规定，即：按照学校要求提交论文和相关材料的印刷本和电子版本；同意学校保留毕业论文的复印件和电子版本，允许被查阅和借阅；学校可以采用影印、缩印或其他复制手段保存毕业论文，可以公布其中的全部或部分内容。

以上承诺的法律结果将完全由本人承担！

作 者 签 名：关浔宇

2023 年 **05** 月 **23** 日

摘　　要

随着网易云音乐的发展，越来越多的人在网易云音乐社交平台中交流人生经历和感悟，产生了情感色彩鲜明的海量中文评论。因此，本文围绕网易云歌单评论，分别采用基于情感词典、情感词典与机器学习算法相结合及 TextCNN 等方法，展开中文评论文本的情感极性分析研究。

通过本次实践验证可知，虽然机器学习方法弥补了情感词典方法不擅长处理未定义情感词的缺陷，但是此类方法需在已知结果标签的前提下，才能保证分类性能，无监督聚类算法不擅长处理文本情感分类问题。此外，尽管 TextCNN 能够更好地学习评论文本的上下文语义信息，然而在处理小数据集及异常数据时，此类单一神经网络模型的分类效果略逊于，根据逻辑回归或支持向量机算法搭建的文本情感分类模型。

关键词：情感词典；机器学习；TextCNN；情感极性分析

Abstract

With the development of Netease Cloud Music, an increasing number of people are communicating their life experiences and insights on its social platform, generating a massive amount of Chinese comments with vivid emotional expressions. Therefore, this article focuses on the sentiment polarity analysis of Chinese comment texts on Netease Cloud music playlists, employing methods including sentiment dictionary, combined sentiment dictionary with machine learning algorithms, and TextCNN.

The experimental results show that while machine learning methods make up for the deficiency of sentiment lexicons in handling undefined emotion words, such methods require known result labels to ensure classification performance, and unsupervised clustering algorithms are ineffective at text sentiment classification. Moreover, although TextCNN can better learn contextual semantic information of comment texts, its performance is slightly inferior to text sentiment classification models built using logistic regression or support vector machines when dealing with small datasets or exceptional data.

Key Words: Sentiment Polarity Analysis; Sentiment Dictionary; Machine Learning; TextCNN

目 录

第1章 绪 论 1
1.1 研究背景与意义 1
1.1.1 研究背景 1
1.1.2 研究意义 1
1.2 研究现状 2
1.2.1 国内研究现状 2
1.2.2 国外研究现状 2
1.3 主要研究内容 2
1.3.1 数据采集 3
1.3.2 数据预处理 3
1.3.3 基于情感词典的中文情感识别 3
1.3.4 基于聚类算法的中文情感识别 3
1.3.5 情感词典与分类算法融合的中文情感识别 3
1.3.6 基于 TextCNN 的中文情感识别 3
1.4 论文组织结构 4
第2章 核心技术介绍 5
2.1 数据采集 5
2.2 文本预处理 5
2.2.1 文本分词 5
2.2.2 词性标注 5
2.2.3 文本表示模型 6
2.3 情感词典 6
2.4 文本分类算法 6
2.4.1 有监督分类算法 6
2.4.2 无监督聚类算法 8
2.4.3 TextCNN 9
2.5 模型评估方法 10
第3章 数据采集 11
3.1 系统流程 11
3.2 采集工具 11
3.3 参数采集 12
3.4 评论爬取与存储 12

第 4 章 基于情感词典的文本情感分析 13
4.1 系统流程 13
4.2 文本分词 13
4.3 词性标注 14
4.4 情感极性计算 15
第 5 章 基于无监督聚类算法的文本情感分析 17
5.1 系统流程 17
5.2 数据预处理 17
5.3 模型训练及评估 18
5.4 实验结果分析 19
第 6 章 情感词典与分类算法融合的文本情感分析 21
6.1 系统流程 21
6.2 模型训练及评估 21
6.2.1 逻辑回归 21
6.2.2 支持向量机 22
6.2.3 KNN 22
6.2.4 朴素贝叶斯 22
6.2.5 随机森林 23
6.3 实验结果分析 23
第 7 章 基于 TextCNN 的文本情感分析 25
7.1 系统流程 25
7.2 构建 Vocal 词典 26
7.3 CNN_Classifier 的实现 27
7.4 数据加载 28
7.5 模型训练及评估 28
7.5.1 DeepModel 初始化 29
7.5.2 训练评估方法 29
7.5.3 性能度量方法 30
7.6 实验结果分析 31
第 8 章 总结与展望 33
致　　谢 35
参考文献 37

第 1 章 绪 论

1.1 研究背景与意义

1.1.1 研究背景

近年来，文本情感识别研究受到了广泛关注，更是出现了 ChatGPT 这样能够模拟人类聊天行为的问答系统。从技术理论层面出发，文本情感分析的研究需要涵盖自然语言处理、机器学习以及深度学习等专业领域的知识。迄今为止，文本情绪判别方法通常分为情感词典、机器学习和深度学习三类。

其中，使用深度学习方法处理文本情感分类任务，是当下主流的研究方向。虽然深度学习方法能够使用类似于 N-Gram 元素的原理，更好地考虑文本内部语义，克服传统方法中忽略上下文语义的问题。但是此类方法在处理语义结构较为复杂的中文文本时，易出现模型训练时间较长、过拟合及缺乏可解释性等问题，模型性能往往不如传统的情感识别方法。针对于网易云歌单评论而言，在选用较为高效的情感词典的前提下，情感词典的方法具有准确率高、易于实现及不依赖情感极性标签等优点。而使用机器学习方法分析文本情感类型时，若给定的数据集无结果标志，则只能使用无监督的聚类算法 KMeans；若给定的文本数据集有标签，则可以使用诸如逻辑回归等有监督分类算法。大量研究证明，即使在中文文本数据集体量比较小的情况下，使用机器学习方法进行文本情感评估实践，仍能得到较理想的分类效果。

1.1.2 研究意义

随着互联网时代的飞速发展，人际交流愈加便捷快速，随之也产生了饱含丰富人文情感的海量信息。因此，使用人工智能技术对网络上存在的中文文本展开情感识别研究，将在社会化媒体、舆情监测及智慧客服等领域，展现极大的商业应用价值。

此外，采用多种文本情感极性分析方法，针对于网易云歌单评论展开情感识别研究实践。不仅能够挖掘用户对于具体网易云歌单的情感倾向，从而筛选出定位明确、畅听体验极佳的网易云歌单。还能根据用户的情绪反馈，将用户聚类划分为若干类别，实现类似歌单或歌曲的精准推荐，极大提高用户粘性。

1.2 研究现状

1.2.1 国内研究现状

近年来，人工智能发展战略连续三年写入政府工作报告，中国政府更是规划了北京、上海、深圳、成都、合肥及济南等城市作为人工智能产业发展先行试点区。依托国家政策风向，自然语言处理相关的研究方向也因此在国内呈现出蓬勃发展的趋势。针对于中文情感识别研究领域的国内研究现状，王海宁[1]指出：“目前，国内人工智能领域的专业人士，主要从字词、句子及话题等级别出发，展开情感识别研究工作”。王婷等人[3]就文本情感极性分析方法的发展现状，指出：“目前此类研究主要有：情感词典、机器学习及深度学习三类方法。其中，深度学习方法可细分为以下四类：单一神经网络模型、融合网络模型、引入注意力机制的网络模型及使用预训练词向量模型的情感分析技术。”在基于机器学习的文本情感分类领域，刘丹丹等人[7]使用支持向量机的方法，针对微博评论进行了情绪判定研究，解决了情感词典方法，无法处理未定义情感词的问题；王志颖等人[10]基于 Spark 和传统机器学习方法展开了文本情感分类研究，解决了传统方法，无法高效地实时处理格式多样的海量数据这一问题。在基于深度学习的文本情感分类领域中，木欣喜等人[15]使用引入注意力机制的 Bi-LSTM 方法展开了网评情感判别研究，解决了单一网络模型方法处理文本语义关注度低下，导致模型准确率较低的问题。

1.2.2 国外研究现状

随着大数据、云计算及物联网等技术的飞速发展，文本情感识别研究在国外，正在以惊人的速度发展。Hung Lai Po 等人[16]从文本情感分析的发展现状出发，论述了基于文本的情感分析最新趋势。迄今为止，基于深度学习方法的情感识别研究有了显著进展，已成为文本情感判定领域的主流研究方向。相较于传统情感分类技术，Yao Lisha 等人[17]从基于深度学习的文本情感分析方法出发，采用 CNN-LSTM 混合神经网络展开了酒店评论的情绪分析，解决了传统情感分析方法无法识别处理文本隐含语义与上下文关联问题的同时，也一定程度上缓解了深度学习方法在处理复杂语义结构文本时易发生过拟合现象及模型缺乏可解释性的问题。此外，Jyostna Devi Bodapati 等人[18]从深度学习与元学习方法的角度出发，展望了文本情感识别研究领域未来的发展方向。

1.3 主要研究内容

本文选用网易云音乐评论为研究对象，分别采用情感词典、无监督聚类算法、情感词典与有监督分类算法融合及 TextCNN 方法，展开情感分类研究。研究内容如下：

1.3.1 数据采集

针对网易云音乐 API 数据接口，采用静态网页采集技术，获取相应的 html 文本信息。并依赖字典索引的方式，从转换为 json 格式的文本数据中，抽取出所有的评论文本，按行存储为 CSV 格式文件。

1.3.2 数据预处理

在实践过程中，需要预先清洗掉评论样本中冗余的字符，并将评论文本转化为与多种分类算法实例化过程中输入数据格式要求相匹配的数据集。

1.3.3 基于情感词典的中文情感识别

依赖 BosonNLP 情感词典，使用文本分词、情感词抽取及情感极性评分等技术，实现中文评论文本的情感分类任务，并将原始的评论样本打上情感极性标签，达到为有监督分类算法实例化提供标签支持的目的。

1.3.4 基于聚类算法的中文情感识别

KMeans 作为一种无监督机器学习聚类算法，无需事先通过情感词典的方法将评论样本打上情感极性标签，就能完成中文情感识别研究工作。因此，基于 KMeans 无监督聚类算法完成文本情感极性研究，可以作为基于情感词典与有监督机器学习算法相结合的方法的对照组，对比分析出后者的优势之处。

1.3.5 情感词典与分类算法融合的中文情感识别

采用支持向量机、逻辑回归、随机森林、朴素贝叶斯及 K 最近邻算法，展开基于情感词典与有监督分类算法相结合的中文情感识别研究，分析对比多种机器学习分类算法在处理文本情感分类问题时的泛化性能。

1.3.6 基于 TextCNN 的中文情感识别

深度学习方法，作为当下主流的研究方向。即使研究表明，单一网络模型不擅长处理中文情感极性分类任务。作为毕业设计，仍需要使用 TextCNN 算法展开情感识别研究实践，验证分析卷积神经网络算法在文本情感分类领域的优缺点。

1.4 论文组织结构

根据前文研究思路和内容，论文组织结构如下：

第一章 绪论：综合概述中文情感识别研究领域的主流技术理论及优缺点，并给出本文的研究方向与思路。

第二章 核心技术介绍：介绍涉及到的相关技术理论，包含数据采集与预处理、机器学习分类算法及 TextCNN 算法等。

第三章 数据采集：介绍采集网易云歌单评论的具体过程及存储方式。

第四章 基于情感词典的文本情感分析：介绍具体的实践过程及实验目的。

第五章 基于无监督聚类算法的文本情感分析：介绍具体的实践过程，作为实验对照组，给出适宜的结论。

第六章 情感词典与分类算法融合的文本情感分析：介绍具体的实践过程，比较文本情感分类任务中不同算法的实际效果，给出适宜的结论。

第七章 基于 TextCNN 的文本情感分析：介绍具体的实践过程，对比机器学习方法的文本情感分类效果，给出适宜的结论。

第八章 总结与展望：总结本次毕业设计学习与实践的内容，展望针对于文本情感识别研究领域，未来的学习与实践期望。

第 2 章　核心技术介绍

2.1 数据采集

网易云音乐网页客户端是基于动态网页原理的信息加载模式，一般通过 JS 脚本实现信息的动态加载，且基于 Ajax 技术实现信息传输。因此，其需要提供经加密算法 AES 构造的两个随机参数 encText 和 encSecKey 来构造 post 表单数据，向 Web 服务器发送请求，才能获得评论信息。但是通过这种模拟用户点击行为来获取评论信息的爬虫程序，需要实现对经 AES 算法加密的参数构造，然后基于此向 Web 服务器发送请求，最终获得相应信息。显然这样的爬虫程序，较为复杂且性能低下。

网易云音乐官方提供了专门的 API 数据传输接口，并发布了相应的开发者使用文档。因此，基于官方提供的 API 及其开发文档，就可以通过基于歌单 id 构造而来的特殊 URL，便捷的实现对歌单中所有歌曲基本信息的采集。这样的爬虫程序，显然更加简单明了。

2.2 文本预处理

文本预处理是一种针对文本数据集通过清洗、转换及归一化等操作，最终获得情感分类模型所需的文本格式及内容的预处理技术。接下来主要介绍，本文所涉及的文本预处理相关工作的原理及过程。

2.2.1 文本分词

文本分词是文本预处理环节的第一步，主要任务是将文本切分为具有特定含义的字词序列，主流分词方法如下。

基于词典的分词方法，主要通过词典匹配来确认句子的词边界。此类方法核心工具是词典，其一般可通过人工标注或深度学习方法生成。在基于词典的中文分词实践过程中，首先要加载采用哈希表实现存储的词典；然后，依序扫描待分词句子并完成最长词匹配；最后，重复前述步骤完成所有待分词句子的扫描匹配处理，以实现中文分词。而基于统计分词的核心思想是，通过海量语料的学习统计，以建立统计模型来实现文本分词，具体方法有最大匹配法等。

2.2.2 词性标注

文本分词处理结束后，一般需要针对分词展开词性标注工作，目标是将词性信息分配标注给相应字词。词性标注的方法主要有：基于规则或基于统计的方法。其中基

于规则的方法要求在特定规则下，对语料进行词性标注。作为情感词典的延伸内容，一般提供有词性词典作为规则模型，以展开词性标注工作。该类方法相较于基于统计的方法，具有标注过程简单易懂的特点。

2.2.3 文本表示模型

文本表示模型是一种将文本映射为向量的技术。在文本情感分类任务中，模型无法直接接受自然语言文本作为输入样本。因此，在训练模型的过程中，需要事先将评论文本转化为向量的形式作为输入参数，才能正常进行情感识别研究实践。本文主要采用 TF-IDF 方法，将文本投影为向量序列。

从实现原理的角度出发，此类模型通过 tf-idf 值反映字词在句子中的重要程度，一般认为在句子中频繁出现，但在全局语料库中出现频次较低的字词更具代表性，其 tf-idf 值自然也越高。

从数学原理的角度出发，其首先假设一个由 n 个文本组成的文本集 W={w1,w2,...,wn}，其中每个文本 wi 中包含词汇表 V={v1,v2,...,vn}中的部分词汇。这样就可以基于每个文本 wi 和词汇表中的每个字词 vi，计算字词在该文本中的频率 tf(vi,wi)。此外，为了消除冗余单词对于结果的影响，例如“a”、“the”等，需要计算逆文本频率 idf(vi,W)，其公式如下：

$$idf(v_i, W) = \log \frac{n}{|\{w_j : v_i \in w_j\}|} \tag{2.1}$$

其中，n 是文件集 W 中的文件总数，|{wj:vi∈wj}|指代字词 vi 出现在文件集 W 中的文件数量。因此，对于每个文件 wi 和字词 vi∈V，可将字词 tf-idf 表示为：

$$tfidf(\mathrm{vi}, wi, W) = tf(vi, wi) * idf(vi, W) \tag{2.2}$$

2.3 情感词典

情感词典是一个由字词及其情感极性数值构成的字典，常应用于情感识别研究领域。作为一种计算情感极性的工具，情感词典一般由人工标注、基于语义或基于统计的方法生成。目前主流的研究思路是，使用深度学习方法生成情感词典。在本次实践中，考虑到自定义情感词典的生成难度，选择直接采用 Boson 情感词典作为实验依据。

2.4 文本分类算法

2.4.1 有监督分类算法

在情感词典与有监督分类算法融合的中文情感分析中，使用情感词典方法将评论

文本打上情感极性标签后，以完成含结果标签的文本集构造。并基于此依次使用多种有监督分类算法，展开中文情感分析。涉及算法的具体介绍如下：

1. 逻辑回归

当使用普通线性回归模型，处理因变量离散化问题时，其关于“误差项服从均值为 0 的正态分布”的前提不再成立，误差项对于影响因素 X 而言，不再可以省略不计。而在广义线性模型中，因变量不再要求连续且正态，能够对服从正态分布、二项分布及泊松分布的随机因变量进行建模。因此，逻辑回归作为一种特殊的广义线性模型，y 不再仅仅是服从二项分布的随机因变量，而是模型输出的期望值 E(y)，广义线性模型也基于此消除了误差项不可忽略问题。

二分类因变量逻辑回归模型，实际上表示了输入影响因素向量 X 与输出模型期望 E(y)之间的回归关系，构造逻辑回归模型的关键在于如何对两者进行拟合。一般选用 sigmond 函数作为连接函数 g()，来拟合 X 与 E(y)。其公式如下：

$$g(E(y)) = \ln(p/(1-p)) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p = X\beta \qquad (2.3)$$

其中，p 表示样本预测为反例的概率，X 表示输入影响因素因子，β表示输入影响因素权重。

2. 支持向量机

支持向量机是一种依赖特征空间的最大间隔线性分类器。其根据间距最大化的策略，将实际情景下的分类问题转化为数学逻辑上关于优化凸二次规划的问题。换而言之，SVM 将输入数据视为 n 维特征空间中的样本点，并尝试找到超平面，将不同类别的样本分开。这个超平面的维度是 n-1 维，而在此平面上的点称为支持向量。若应用场景不是完全线性可分的，则可以将高维函数应用于原始数据，从而使用 SVM 进行有效分类。而对于线性可分的情况，SVM 可以简单地画一条超平面，在二维场景下就是一条直线，将正负样本分开。

3. 朴素贝叶斯

朴素贝叶斯算法在特征相互独立的前提下，通过计算后验概率的方式，最终确定样本类别。此类算法虽然简单高效，但是不适用于处理特征之间存在相关性时的情形。具体的算法步骤是：提取训练集特征并转换为特征向量；计算各类别先验概率 P(Ci)，其中 Ci 表示第 i 个类别；计算每个特征在每个类别下的条件概率 P(Xi|Ci)；针对新的样本，计算其后验概率 P(Ci|X1,...,Xn)，并将其划分到最可能的类别中。

4. 随机森林

随机森林算法通过创建多个决策树的方式进行预测，其中每棵决策树都是基于随机特征选择和自助采样技术构建而来。这种引入随机性的方式使得每棵决策树的差异性较大，从而提高了分类器的准确率和泛化能力。在随机森林的具体构建过程中，首先要在训练集中有放回的随机抽样出 n 个样本集，然后每个样本集随机选择 m 个特征

并分别构建决策树模型，最后根据决策树们的投票结果来决定样本类别。在决策树构造过程中的划分时间节点环节，常使用信息增益标准来完成特征选取，公式如下所示。

$$Gain(D,w)=ent(D)-\Sigma\frac{D_1}{D}ent(D_1) \tag{2.4}$$

简而言之，信息增益公式是衡量属性对于决策树划分贡献的核心指标之一，它量化了样本集合及其子集合不确定性程度，并通过计算划分前后信息熵的变化量来决定最佳划分属性。

5. K 最近邻

KNN 是一种有监督学习算法，可用于分类和回归问题。该算法通过计算待分类点与其它点间的距离来选取出最近的 K 个点，并将 K 个点中出现频次最高的类别作为待分类点的预测类别。在实践环节中，一般采用交叉验证法，调试选择最为适宜的 K 值；一般选用欧氏距离公式，作为计算样本点间距离的工具。若假设样本点 A=（x1,y2）及 B=（x2,y2），则欧式距离公式可以表示为：

$$dist(A,B)=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2} \tag{2.5}$$

2.4.2 无监督聚类算法

KMeans 算法是一种无监督机器学习算法，不需要结果标签。它的核心思想是：使用 K 个随机初始质心将数据点分配到最近的簇中，并重新计算每个簇的中心坐标，重复这个步骤直到簇中心不再变化或达预期迭代阈值为止。工作流程，如图 2-1 所示。

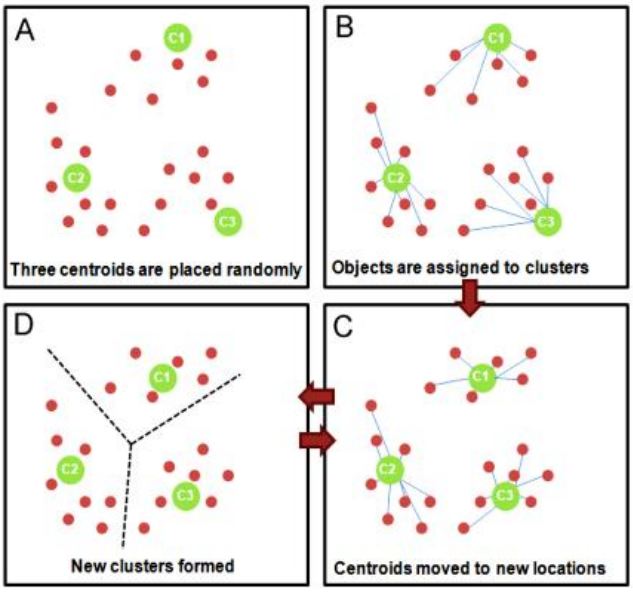

图 2-1 KMeans 算法工作流程

此外，KMeans 算法同 KNN 算法类似，都可以选择欧氏距离作为样本点之间距离的度量方法，且对于选取最优 K 值的方法也如出一辙。

2.4.3 TextCNN

TextCNN 是一种基于卷积神经网络的深度学习算法，一般通过卷积和池化操作，从句子中提取不同的文本特征，并输出外接 softmax 以完成 n 分类任务。

从算法工作流程的角度出发。首先，对句子中的分词进行词嵌入转化。然后，对每个句子的特征图，根据窗口尺寸{2,3,4}，依次挂载多个卷积核，实现不同级别的卷积计算。此外，将卷积处理后的特征图输入到最大池化层中，提取最明显的特征值。最后，将池化层中生成的所有结果输入到全连接层并外接 softmax 函数以完成文本分类任务。

相较于传统的 CNN 网络结构，TextCNN 在结构上更为简单明了。如图 2-2 所示。TextCNN 实际上，只有一层卷积层及一层最大池化层。在卷积层中，一般基于尺寸为 2、3、4 的窗口分别挂载不同的卷积核。在池化层中，使用适宜的池化方法，从卷积层输出的特征图中提取出典型特征。在全连接层中，将所有特征向量拼接成一个定长的一维特征向量，并外接 softmax 函数完成基于典型特征值的文本情感二分类输出。

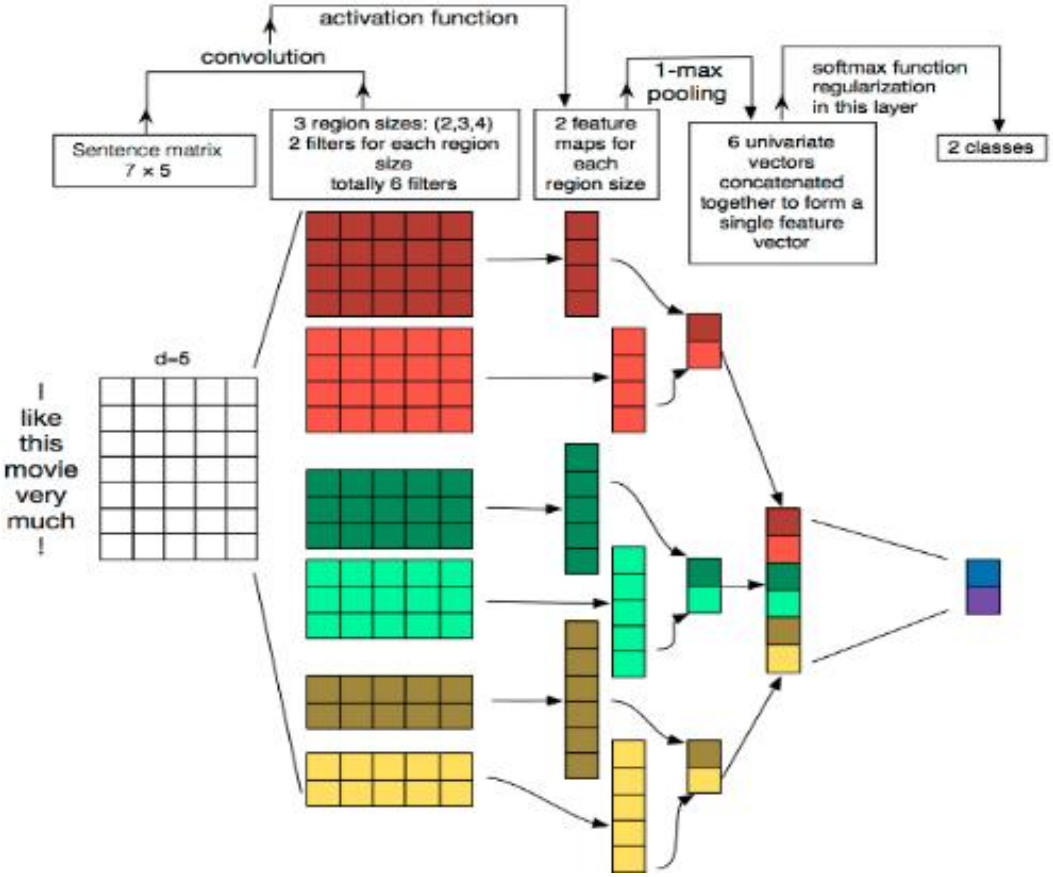

图 2-2 TextCNN 网络结构

2.5 模型评估方法

在中文情感识别研究中，针对模型性能的度量指标，有准确率、精确率及召回率等。本文选用准确率，作为模型泛化性能的度量指标。而在理解准确率之前，需要预先明确混淆矩阵的概念。假设在一个二分类问题中，样本有正负两类，那么预测结果和真实标签的组合就有四种：FP、TP、FN、TN，分别表示：实际为负预测为正、实际为正预测为正、实际为正预测为负、实际为负预测为负。混淆矩阵，如表 2-1 所示。

表 2-1 混淆矩阵

真实情况	预测结果	
	正例	反例
正例	TP（真正例）	FN（假反例）
反例	FP（假正例）	TN（真反例）

因此，准确率表示了有多少比例的样本被预测正确，其公式如 2.6 所示。

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

第 3 章　数据采集

3.1 系统流程

本文基于网易云音乐 NodeJS 版 API，依赖预先采集到的歌单中所有歌曲的 ID 标识，构造出含有特殊参数的 URL，从而实现指定歌单中所有歌曲评论的采集工作。此外，将采集到的评论文本按行存储为 CSV 格式文件，以便于后续工作的集中处理。本环节的工作流程，如图 3-1 所示。

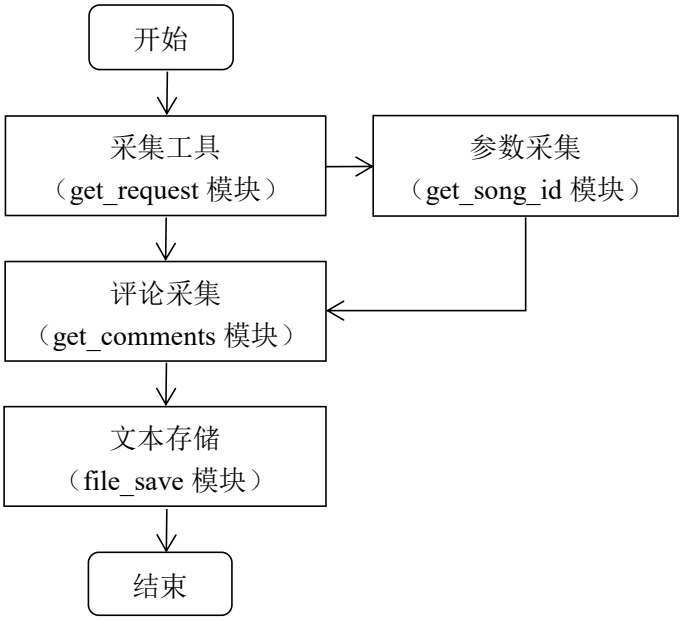

图 3-1 数据采集系统工作流程

3.2 采集工具

本环节基于网易云音乐 API 及 requests 模块，完成数据采集工具 get_request 模块的开发，实现了歌曲基本信息及评论的 HTML 文本采集。

在 get_request 模块中，主要通过 requests 模块完成 HTTP 请求的发送，并且使用 requests.session().get()方法实现数据流传输过程中 cookies 和 session 状态的管理，具体代码如下：

```
requests.session().get('http://music.163.com/playlist?id=%s' % a, headers=headers)
```

3.3 参数采集

本环节主要依赖数据采集工具 get_request 模块，通过文本字符串正则匹配的方式，实现了歌单中所有歌曲 ID 的参数采集，为后续文本采集工作做好了关于歌曲 ID 的参数准备。其中，关于文本字符串正则匹配的实现方式如下：

```
re.findall(r'<a href="/song\?id=(\d+)">(.*?)</a>', response)
```

3.4 评论爬取与存储

本环节根据歌曲 ID 列表 song_id_list，依次完成相应歌曲评论的爬取与追加存储。针对于评论文本爬取，首先需要依赖 get_request 模块完成相应 HTML 文本的爬取，然后需要将 HTML 文本转化为 json 格式文本，之后通过类似字典键值索引的方式提取评论文本信息。实现代码如下：

```
html = get_request(new_url, 0)
json_dict = json.loads(html)
comments = json_dict['comments']
for item in comments:
    commentlist.append(item['content'])
```

最后，将采集而来的 19977 条评论文本，按行写入到 CSV 表格文件中。网易云音乐中文评论数据集 comments.csv，如图 3-2 所示。

1	1	对于男人来说阳刚其实是个褒义词
2	2	我缺少女朋友能不能给我找一个？[大笑]
3	3	In the end
4	4	美国和中国不踢足球[强]
5	5	笑死了
6	6	了解规则练习规则熟悉规则打破规则创造规则
7	7	高二的至暗时刻靠的就是这首歌，，
8	8	真香
9	9	很勇敢的人，那岂不是比表面有所谓阳刚之气的人厉害？
10	10	性取向和性格不应该被人定义，无论是男性还是女性，只要勇敢 有责任 无论生理性别是内心性别
11	11	好听，美丽，魔怔了
12	12	好听，美丽，魔怔了
13	13	哈哈哈哈好可爱[开心]
14	14	然后b就问“你多大”（b就是我回复的那个人）
15	15	就是a说“防沉迷系统都没有考虑过小学生，一个小时最后只有55分钟巴拉巴拉”
16	16	说啥了？好奇
17	17	肯定还没有小升初哈哈哈哈。看他的发言就知道
18	18	in the end 今天英语课刚学了这个词，译为最后，没想到今天学了 中午刚刚好就遇到了[吐舌]
19	19	一听到这个歌我就想到神盾局斯凯的剪辑
20	20	哥们你厨艺高吗？
21	21	阳刚之气随着时代进步应该改一改说法，我见过行为举止gaygay长相阴柔的男孩子也很有血性，有
22	22	你多大？
23	23	一切都会过去的
24	24	[流泪]
25	25	只有相信自己的人能走到最后

图 3-2 中文评论数据集（节选）

第 4 章　基于情感词典的文本情感分析

4.1 系统流程

在基于情感词典的文本情感分析实践中，计算出评论样本的情感极性分数，完成评论文本的情感二分类任务，从而解决后续分类算法在进行文本情感分类时，需要含结果标签的文本数据集问题。本环节实践的系统流程，如图 4-1 所示。

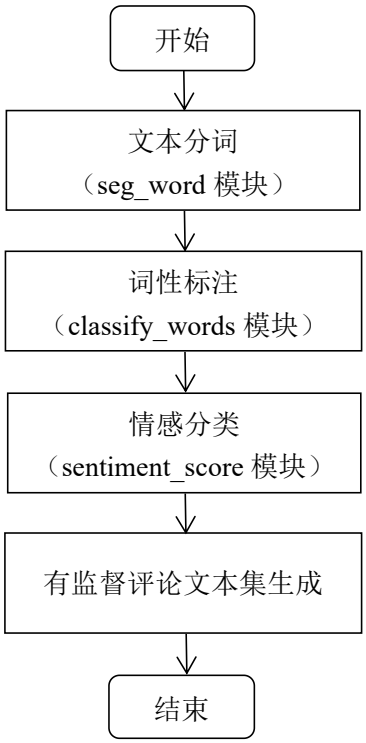

图 4-1 基于情感词典方法的系统流程

4.2 文本分词

本环节主要使用停用词、否定词及程度副词词典及 jieba 完成文本分词模块 seg_word 的开发。其中所涉及的词典，来源于专门提供用于 AI 训练数据的数据堂网站（https://www.datatang.com/）。

在实践中，除了考虑情感词外，还需考虑否定词和程度副词对于文本情感类别结果的影响。因此，需根据否定词及程度副词词典，清洗掉停用词典中的否定词及程度副词，从而防止遗漏此类信息。在项目实践的过程中，需要使用 open()方法实现文本的读写操作，并通过遍历停用词的方法，依次判定是否为否定词或程度副词，并将不满足相应条件的停用词写入到 stopwords.txt 文件中。实现方式如下：

```
with open(stopwords_path, 'w', encoding='utf-8') as f:
```

```
for i in stopwords:
    if (i not in not_word_list) and (i not in degree_list):
        f.write(word + '\n')
```

而在使用 jieba 进行文本分词时。首先，需要通过 open()方法导入停用词词典对象，形成停用词列表。然后，将待分词文本传入分词器 jieba.cut()中进行分词，获得分词结果。最后，需遍历分词结果并清洗掉其中的停用词，实现去除停用词的目的，实现方法如下：

```
list(filter(lambda x: x not in stopwords, seg_result))
```

4.3 词性标注

词性标注模块 classify_words，应具有标注文本中情感词、否定词及程度副词的能力。其中，情感词表达了特定的情感倾向，否定词可以反转情感词的情感倾向，程度副词可以改变情感词的情绪表达程度。

在程序实践的过程中：首先，需要读取情感词、否定词及程度副词词典。以情感词词典为例，词典内容如图 4-2 所示。

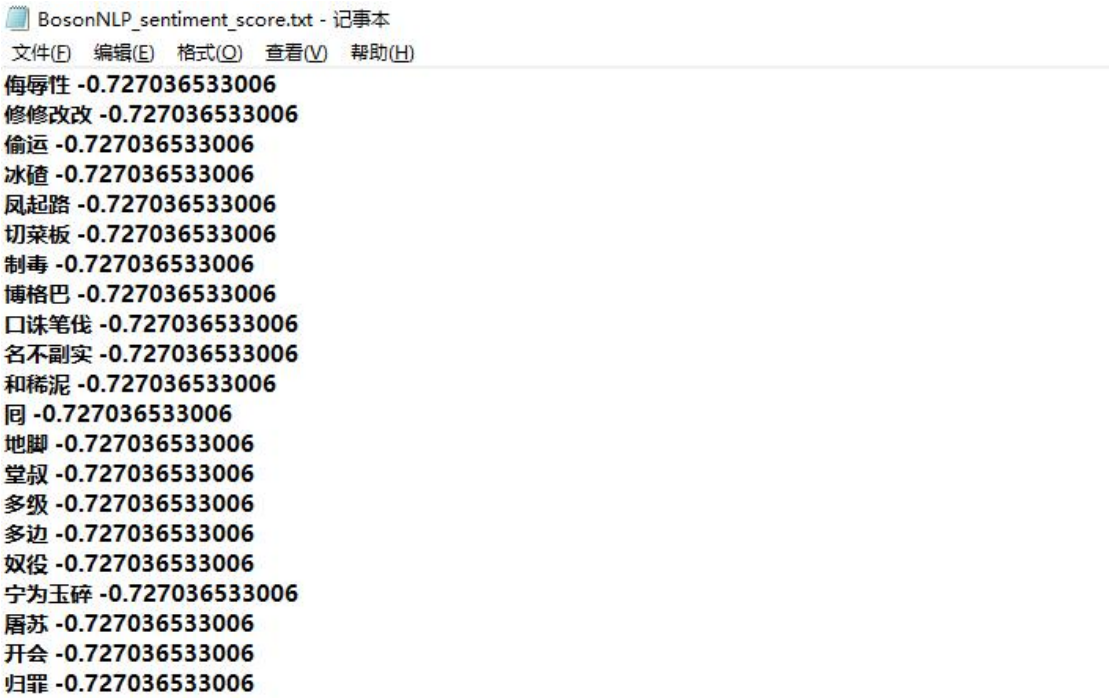

图 4-2 情感词典（节选）

然后，根据上述词性词典，识别句子中分词的词性，并将分词在句子中的位序值，存储到情感词、否定词或程度副词字典对象中。在情感词及否定词字典对象中，每个词位序键索引都对应了情感或程度副词词典中相应字词的情感极性指数。在否定词字典对象中，每一个词索引所对应的情感极性值都可设置为-1，用以表示否定词的情感极性翻转特性。实现方式如下：

```
if w in sen_dict.keys() and w not in not_word_list and w not in degree_dict.keys():
    sen_word[i] = sen_dict[w]
elif w in not_word_list and w not in degree_dict.keys():
    not_word[i] = -1
elif w in degree_dict.keys():
    degree_word[i] = degree_dict[w]
```

4.4 情感极性计算

在 score_sentiment 模块中，将前述词性标注环节输出的情感字典对象 sen_word、否定词字典对象 not_word 及程度副词字典对象 degree_word，作为本模块的主要输入对象；将文本分词环节输出的评论样本分词对象 sen_word 作为补充输入对象。实现文本情感极性计算的代码逻辑如下：

1. 初始化得分 score 为 0，情感词索引 sentiment_index 为-1。并取出情感字典对象 sen_word 中所有词位序索引，另存为 sentiment_index_list 列表对象。

2. 从 0 开始遍历词位序，若判断为情感词：根据其在情感字典对象 sen_word 中相应位序的情感系数，乘以权重 w（初始为 1），累加更新得分 score。此外，还需要移动一位情感词索引 sentiment_index。

3. 若当前位序字词为情感词且不是最后一个时：则需基于 sentiment_index 遍历当前情感词及后一个情感词之间所有的 seg_result 词位序。若存在否定词，则权重更新为-1，若存在程度副词，则权重更新为相应的程度系数。使得下一个情感词计分时，引入否定词及程度副词的情感极性干涉。

score_sentiment 模块的实现方式如下：

```
def score_sentiment(sen_word, not_word, degree_word, seg_result):
    W = 1
    score = 0
    sentiment_index = -1
    sentiment_index_list = list(sen_word.keys())
    for i in range(0, len(seg_result)):
        if i in sen_word.keys():
            score += W * float(sen_word[i])
            sentiment_index += 1
            if sentiment_index < len(sentiment_index_list) - 1:
                start = sentiment_index_list[sentiment_index]
                end = sentiment_index_list[sentiment_index + 1]
```

```
            for j in range(start, end):
                if j in not_word.keys():
                    W *= -1
                elif j in degree_word.keys():
                    W *= float(degree_word[j])
        if sentiment_index < len(sentiment_index_list) - 1:
            i = sentiment_index_list[sentiment_index + 1]
    return score
```

经过上述 score_sentiment 模块的处理，每一个评论样本都将计算出对应的情感极性指数。由于基于情感词典的方法实践的主要目的是：给评论文本打上情感极性分类标签。因此，需要标定得分不大于零的评论样本为类别 1，其它样本标定为类别 2，从而实现情感极性二分类。最后，需要将二分类标签覆盖写入到 comments.csv 文件中的索引列，从而生成含分类结果标签的 new_comments.csv 文件。new_comments.csv 文件的部分内容，如图 4-3 所示。

353	2	谢谢
354	2	原创版是查斯特的肉体在呐喊，这个版本更像是他的灵魂在吟唱
355	1	我忍这个四楼广告很久了［怒］
356	1	梦该醒了，这里不是拉特兰，向同胞举铳并不会被铳排斥，更不会直接被逐出国门
357	2	想到了赤瞳的种种
358	1	老查.....我们想你了
359	1	挺你，清醒
360	2	笑死我了哈哈哈哈
361	2	还行吧 我觉得填词还起点睛作用了
362	2	真香
363	1	洋文而已
364	2	不要独白说唱啊。。。。破坏气氛
365	1	你女朋友肯定也被绿过很多次了
366	1	嘴上说着不要，身体却还挺诚实

图 4-3 含分类结果标签的评论文本集（节选）

第 5 章　基于无监督聚类算法的文本情感分析

5.1 系统流程

KMeans 算法作为一种无监督聚类算法，摆脱了有监督机器学习算法在处理文本情感分类问题时，对于含分类结果标签数据集的依赖。因此，本环节从 KMeans 算法出发，展开基于无监督聚类算法的文本情感分析实践，并根据实验结果给出相应的分析结论。系统流程如图 5-1 所示。

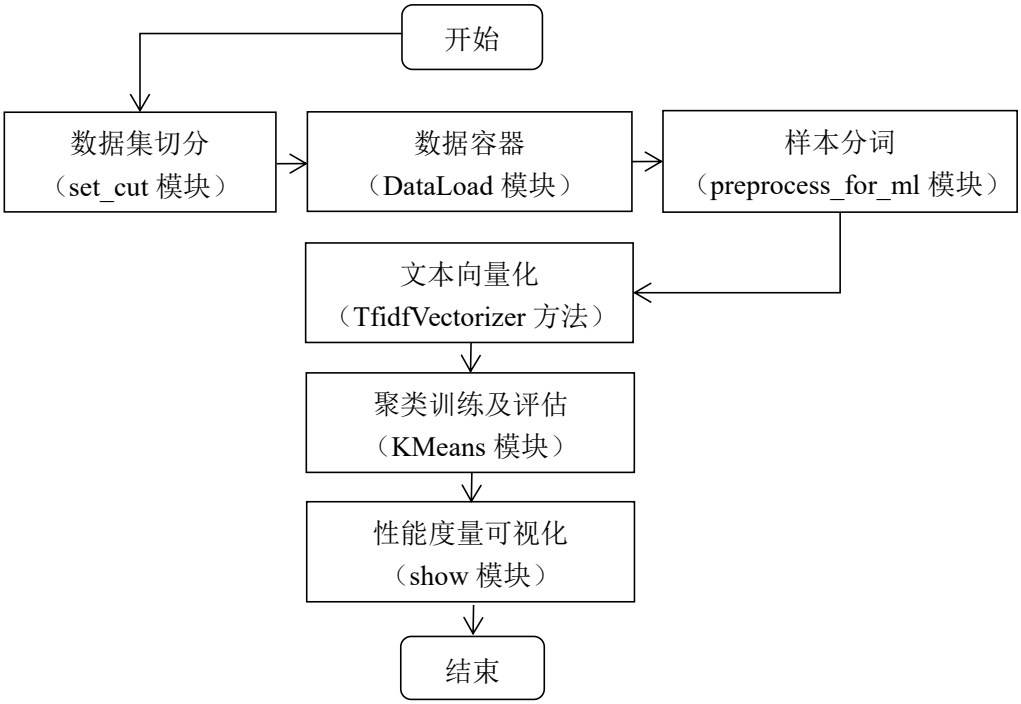

图 5-1 基于 KMeans 算法的文本情感分类系统流程

5.2 数据预处理

在正式进行模型训练与预测之前，需要事先完成数据预处理操作，以保证后续环节的正常进行，具体实现步骤如下：

首先，把数据集按照 20：1 的比例划分为训练及预测集，并分别存储为 CSV 格式文件。切分数据集的方法如下：

train_test_split(data_set, test_size=0.05, random_state=42)

然后，自定义“[(label, sentence), ...]”格式的数据结构类 DataLoad，用来存储和组织训练集及预测集对象。在 DataLoad 类中，首先会按行读取评论样本；然后根据逗号将分类标签与评论划分开来，并转换为“(label, sentence)”类型的元组对象；最后，通过列表集中存储这些元组对象。DataLoad 类的代码如下：

```
class DataLoad(Dataset):
    def __init__(self, split, data_dir="./datasets/"):
        assert split in ["train", "test"]
        self.split = split
        self.data_dir = data_dir
        self.pairs = self.load_data()
    def load_data(self):
        pairs = []
        with open(join(self.data_dir, self.split+".csv"), errors='ignore') as f:
            for line in f:
                label, sentence = line.split(",", 1)
                sentence = sentence.strip("\n")
                label = int(label) - 1
                pairs.append((label, sentence))
        return pairs
```

最后，通过字词的 tf-idf 值作为特征，将评论文本进行向量化处理。以文本“非常棒”及“好”为例，前者的向量化表示为：[tfidf(非常),tfidf(棒),0]，后者为：[0,0,tfidf(好)]。实现方式如下：

```
TfidfVectorizer(token_pattern=r"(?u)\b\w+\b").fit_transform(train_sents)
```

5.3 模型训练及评估

在本环节的实践过程中，一般基于训练评论集 train_st、测试评论集 test_st 及测试评论标签 test_lab，使用 sklearn.cluster 模块的 KMeans 方法，完成聚类模型的训练与预测。实现方法如下：

```
Pred = KMeans(n_clusters=2).fit(train_st).predict(test_st)
```

其中，由于需要解决二分类问题，超参 n_clusters 的数值应指定为 2。然后，将预测结果 Pred 与测试集类别标签 test_lab 进行对比，以实现准确率的计算，从而完成模型泛化性能的度量。实现方式如下：

```
np.mean(Pred == np.array(test_lab))
```

5.4 实验结果分析

将事先划分好的预测评论集 test_st，投喂给训练好的 KMeans 模型，从而获得所有测试评论的情感二分类结果 Pred，如图 5-2 所示。

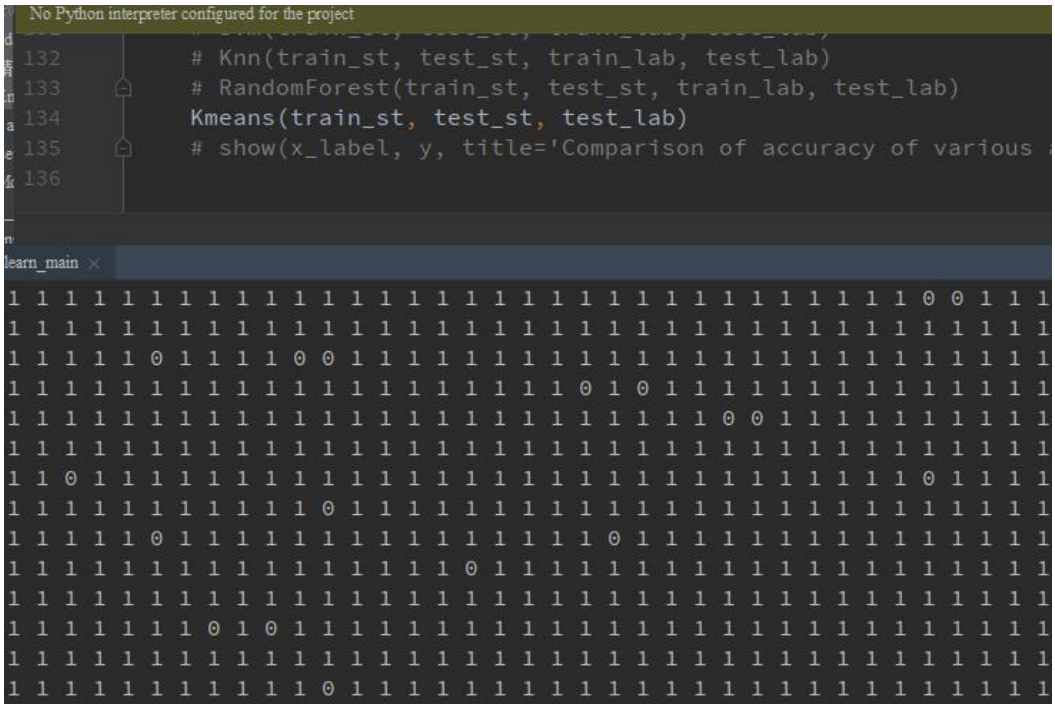

图 5-2 测试集的预测结果 Pred（节选）

然后，通过将预测结果 Pred 与测试集类别标签 test_lab 进行对比的方法，可计算出 KMeans 模型在预测集上的准确率仅为 40.1%。

由此可知，在基于无监督聚类算法的中文情感分类任务中，KMeans 算法尽管摆脱了有监督分类算法对于含分类结果标签的依赖，也一定程度上解决了基于情感词典方法不擅长处理词典中未收录情感词的问题。然而，由于评论文本中情感词和语境的复杂性，KMeans 算法在进行文本向量化和聚类工作时，将丢失大量有效信息，从而导致其在处理中文情感二分类任务时，实际效果不佳。

第 6 章　情感词典与分类算法融合的文本情感分析

6.1 系统流程

在本环节的实践过程中，需要预先通过情感词典的方法，完成文本的情感标签标识工作，然后分别使用逻辑回归、随机森林、支持向量机、朴素贝叶斯及 K 最近邻算法，完成中文情感分类研究实践，对比不同算法的情感分类效果，并简要分析原因。

系统流程，如图 6-1 所示。

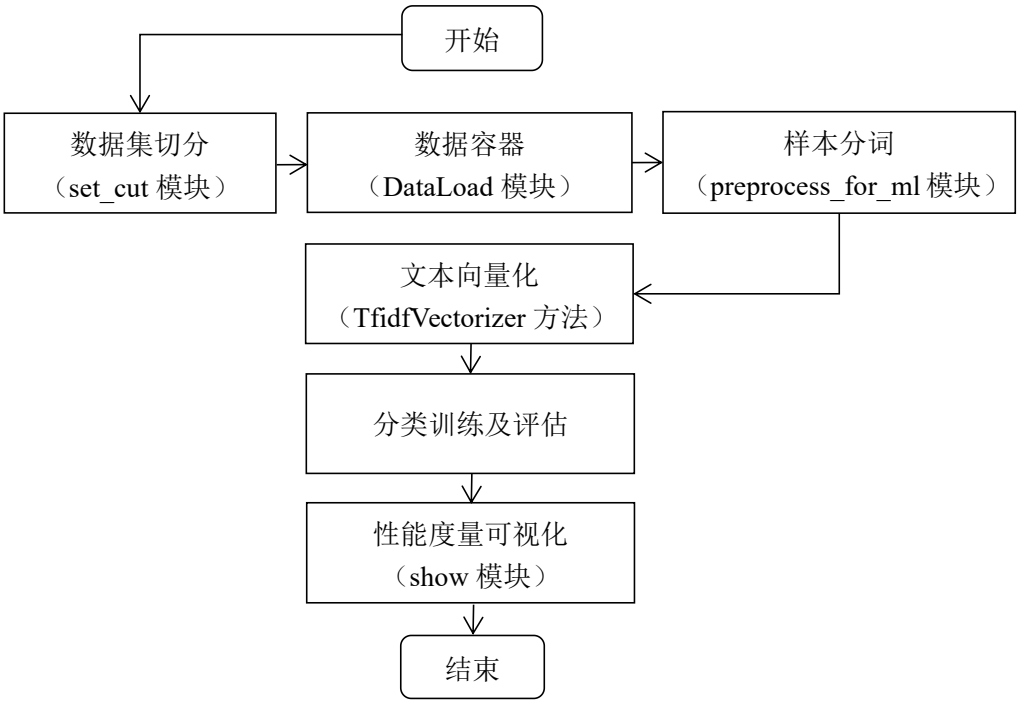

图 6-1 情感词典与分类算法融合的文本情感分类系统流程

6.2 模型训练及评估

为了保证多种模型性能对比结果的正确性，需要保证数据预处理及评估方法的一致性。因此，本环节的数据预处理及评估操作，同前述基于无监督聚类算法的文本情感分析实践中的预处理及评估过程一致，此处不再重复叙述。

本环节基于情感词典方法获取的含分类结果的评论数据集，分别采用多种有监督机器学习算法展开文本情感分类模型的训练与预测，具体过程如下。

6.2.1 逻辑回归

根据逻辑回归算法构建文本情绪分类模型时，一般基于训练评论集 train_st、测试评论集 test_st、训练评论标签 train_st 及测试评论标签 test_lab，使用 sklearn.linear_model

模块的 LogisticRegression 方法，完成聚类模型的训练及预测应用。实现方法如下：

LogRegression(train_st, test_st, train_lab, test_lab, "lbfgs", 3000).predict(test_st)

在建模的过程中，需要重点关注优化算法 solver 及最大迭代次数 max_iter 等超参的选择。本文选用 lbfgs 作为逻辑回归模型建模过程中的优化算法，相较于其它优化算法，lbfgs 使用了类似随机梯度下降的技术，可以更高效且稳定地处理大规模数据集。针对于模型最大迭代次数 max_iter，选择设置为 3000 次，这样即保证了模型训练的时间较短，又能极大发挥模型的学习能力。

6.2.2 支持向量机

根据支持向量机算法，构建文本情绪分类模型时，本文直接选用了分类器 SGDClassifier，完成了基于支持向量机的模型训练及预测应用。实现方法如下：

SGDClassifier(max_iter=1000, tol=1e-3).fit(train_st, train_lab).predict(test_st)

SGDClassifier 作为一种使用 SGD 进行优化过的支持向量机算法变种，在建模过程中，为了保证模型学习能力且兼顾运行时间，应选定最大迭代次数 max_iter 为 1000，容忍的优化目标函数变化量阈值为 1e-3。

6.2.3 KNN

根据 K 最近邻算法构建文本情绪分类模型时，基于训练集 train_st 及训练集标签 train_st，使用来自 sklearn.neighbors 模块的 KNeighborsClassifier 方法，完成了基于 K 最近邻算法的模型训练与预测应用。实现方法如下：

KNeighborsClassifier().fit(train_st, train_lab).predict(test_st)

K 最近邻算法作为最简单的分类算法，在建模过程中，没有调参环节。

6.2.4 朴素贝叶斯

根据朴素贝叶斯算法构建文本情绪分类模型时，基于训练集 train_st 及训练集标签 train_st，使用来自 sklearn.naive_bayes 模块的 MultinomialNB 方法，完成了基于朴素贝叶斯算法的模型训练与预测应用。实现方法如下：

MultinomialNB().fit(train_st, train_lab).predict(test_st)

朴素贝叶斯模型是一种概率模型，要求文本特征相互独立。此类模型解释性较强，在模型训练的过程中，没有可调参数。

6.2.5 随机森林

根据随机森林算法构建文本情绪分类模型时，使用来自 sklearn.ensemble 模块的 RandomForestClassifier 方法，完成了基于随机森林算法的模型训练与预测应用。实现方法如下：

```
RandomForestClassifier(n_estimators=20).fit(train_st, train_lab).predict(test_st)
```

随机森林模型依赖随机构建而成的多个决策树，实现输入样本的类别预测投票。在模型训练的过程中，指定了决策树的构建数量参数 n_estimators 为 20，以保证模型学习能力与时间开销的平衡。

6.3 实验结果分析

将评论文本预测集，依次投喂给不同的分类模型，从而获取最终的预测分类结果 Pred。然后，将预测结果 Pred 与测试集类别标签 test_lab 进行对比，以实现准确率的计算，从而完成模型泛化性能的度量。实现方式如下：

```
np.mean(Pred == np.array(test_lab))
```

通过上述方法的计算，可知支持向量机（SVM）、朴素贝叶斯(Naive Bayes)、随机森林（Random Forest）、逻辑回归（Logistic Regression）、K 最近邻（KNN）与 K 均值聚类（KMeans）算法在中文评论文本情感二分类任务中的准确率对比，如表 6-1 所示。

表 6-1 机器学习算法泛化性能对比

	Logistic Regression	SVM	Random Forest	Naive Bayes	KNN	KMeans
Accuracy	81.4%	80.1%	78.7%	74.5%	61.6%	40.1%

由上表可知，逻辑回归及支持向量机算法在处理评论文本情感分类任务时的泛化性能最好，都达到了 80%左右。其主要原因，可总结为以下三种：

1. 适配高维数据

逻辑回归和支持向量机作为线性模型，在高维空间中具有极好的表现力。此类算法可以学习文本特征之间的复杂关系，并且具有适宜的鲁棒性，可以处理不同维度上的噪声数据。而朴素贝叶斯及 KNN 算法在高维空间中，易发生维度灾难现象，从而导致模型泛化性能下降。

2. 非线性分类问题

逻辑回归及支持向量机在构建模型的过程中，都引入了核函数，能够将低维数据

投影到更高维度的特征空间中，从而实现了非线性分类。由于这种特性，此类算法在处理具有复杂结构及语义的中文评论文本时，具有极好的分类表现能力。相比之下，虽然朴素贝叶斯算法在部分非线性分类问题中表现不错，但是其需要获取大量先验知识，反而不利于对复杂语义信息的处理。

3. 模型解释性

逻辑回归算法作为一种基于概率的分类方法，模型结构及样本特征权重简单明了，使得其在模型训练与应用中具有极好的可解性性。而根据间隔最大化求解思想的支持向量机算法，在使用核函数处理非线性分类问题时，一般通过分析向量位置来计算出最佳决策边界。这种模型特性，其本身解释性就极好。相反的是，朴素贝叶斯及 KNN 算法需要在预测时，对先验知识或距离度量进行计算，在这个过程中需要考虑多个维度之间的相互影响，从而导致模型结构较为复杂。此外，随机森林算法由于模型中需要包含多个决策树，使得模型结构同样相对复杂，难以在实际应用时进行简单明了地解释。

总而言之，针对于网易云音乐中文评论的情感二分类任务。由于无监督聚类模型在训练的过程中，丢失了大量文本特征信息，从而导致此类模型不擅长处理中文文本情感分类问题。相较之下，诸如逻辑回归及支持向量机等有监督分类模型，由于其较好的模型解释性、容错性及非线性分类能力，在处理此类任务时的效果更好，准确率都达到了 80%左右。此外，最简单的有监督分类算法 KNN，由于模型训练过程中本身的计算特性，使其在处理富含大量语义特征的中文文本时，易发生维度灾难现象，从而导致实际情感分类效果不佳。

第 7 章　基于 TextCNN 的文本情感分析

7.1 系统流程

TextCNN 网络结构如图 7-1 所示。使用深度学习算法 TextCNN，处理文本情感二分类任务时的系统流程如下：

1. 构建 Vocal 词典作为词向量文件，并结合随机初始化方法生成的词向量矩阵，在网络模型嵌入层中，完成评论文本的向量化表示工作；

2. 使用 CNN_Classifier，在卷积层中依次基于窗口尺寸{3，4，5}，使用不同的过滤器分别进行卷积计算，以获得不同维度的特征图；

3. 接收卷积层输出的特征图，使用最大池化方法降采样特征图中的最大特征值，并分窗口生成池化后的特征向量；

4. 构建全连接层，接受所有经卷积及池化处理的特征向量，并将这些特征向量映射为固定长度的一维特征向量；

5. 在最后的输出层，使用 sigmoid 函数将全连接层的输出转化为取值范围为 0~1 之间的类别概率值，从而得到文本的类别预测结果。

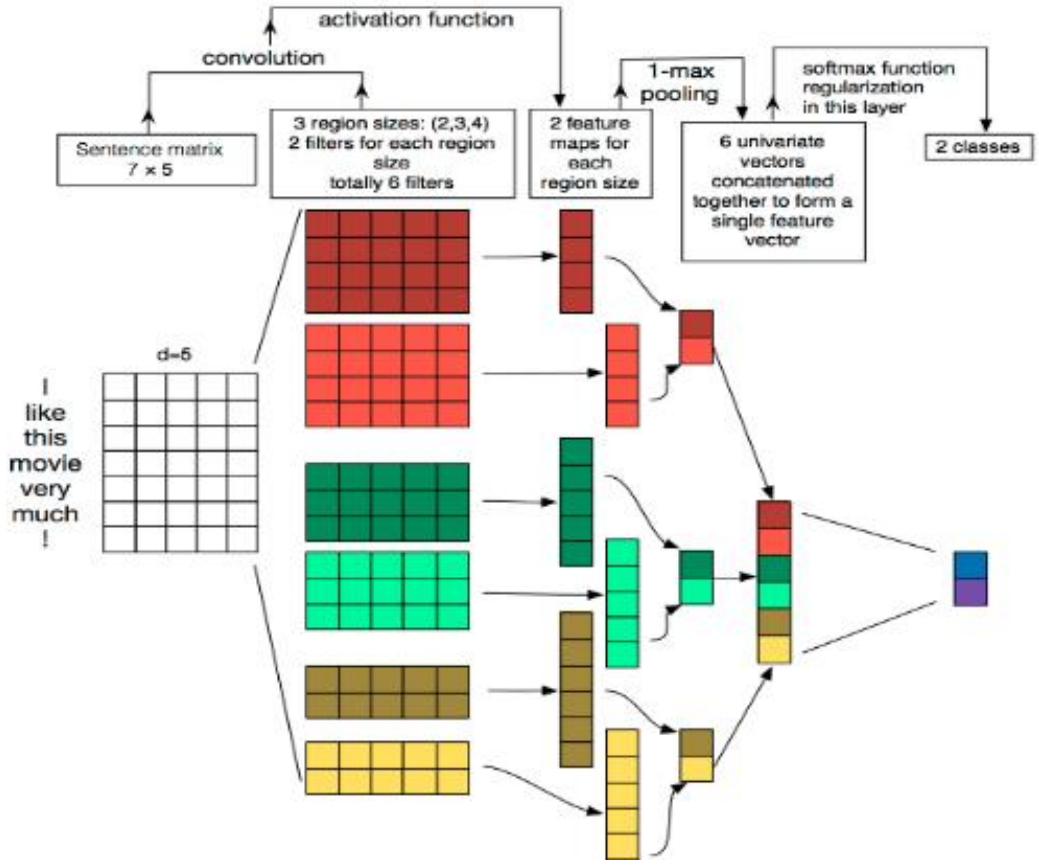

图 7-1 TextCNN 网络结构

7.2 构建 Vocal 词典

由于深度学习模型需要输入数值化数据，因此在使用 TextCNN 算法处理文本分类任务之前，需要生成 Vocal 词典来保存 n-gram 对象与对应整数编号（index）之间的映射关系，从而完成所有评论文本的去重字词与数值索引之间的相互映射，使得神经网络更容易地处理和学习数据。构建 Vocal 词典的流程如下：

1. 通过字符串索引切分的方法，获取所有的 1-gram 及 2-gram 对象；

```
ngrams = [sentence[i:i+self.N] for i in range(len(sentence)-self.N-1)]
```

2. 加载停用词词典，清洗掉 ngrams 列表中的所有停用词；

```
for ngram in ngrams:
    if not set(ngram).intersection(self.stopwords):
        filtered_ngrams.append(ngram)
```

3. 将 1/2-gram 元素存储为字典对象 gram2count 中的键，对应的键值表示为该元素的出现次数，最低为一次；

```
def add_gram(self, ngram):
    if ngram not in self.gram2id:
        self.gram2id[ngram] = self.length
        self.id2gram[self.length] = ngram
        self.gram2count[ngram] = 1
        self.length += 1
    else:
        self.gram2count[ngram] += 1
```

4. 根据字典对象 gram2count 中元素的出现次数，使用 Counter()方法，将所有去重 1/2-gram 对象按照出现次数倒序排序，并按行写入到 vocal.csv 文件中。

```
counter = Counter(voc.gram2count)
with open(path, "a", encoding='utf-8', errors='ignore') as f:
    for word, count in counter.most_common():
        f.write(word + ',' + str(count) + '\n')
```

最终生成的 Vocal 词典内容，如图 7-2 所示。在 vocal.csv 文件中，存在所有评论文本的 1/2-gram 分词元素及其出现频次，共有 28662 条数据。

16	现	1001
17	什	930
18	里	920
19	唱	908
20	开	907
21	情	880
22	感	868
23	喜	858
24	首歌	838
25	十	830
26	前	816

图 7-2 Vocal 词典（节选）

7.3 CNN_Classifier 的实现

在自定义类 CNN_Classifier 中，实现了模型的前向计算方式，并在初始化函数__init__()中定义了诸如嵌入层大小、卷积滤波器数量及窗口尺寸集合等各种超参的默认数值。针对于前向传播函数的实现，其代码逻辑结构可划分为嵌入层、卷积层、池化层及全连接层，这四个层次。具体解释如下：

1. 嵌入层

在嵌入层中，需要引入事先构建的 Vocal 词典作为词向量文件，并结合随机初始化方法生成的词向量矩阵，完成评论文本向量化表示工作。流程如下：

首先，需要使用随机初始化方法，生成词向量矩阵。

torch.nn.Embedding(vocab_size, embed_dim)

然后，引入 Vocal 词典作为词向量文件对象，格式为：{“你”:1, “好”:2...}。

此后，将输入文本分词为 1/2-gram 元素。

```
def get_features(sent):
    unigrams = list(sent)
    bigrams = [sent[i:i+2] for i in range(len(sent)-1)]
    return unigrams + bigrams
```

最后，通过 Vocal 字典获取分词元素的对应索引，再通过索引在词向量矩阵中获取相应的向量表征。这样句子“非常棒”就转化成了向量形式的二维特征矩阵，即{“非”:[0,0,0,0,1],”常”:[0,1,0,0,0],”棒”:[1,0,0,0,1],”非常”:[1,1,0,0,0],”常棒”:[1,0,1,1,1]}。

2. 卷积层

经过嵌入层将评论文本转化为二维矩阵后，卷积层将基于尺寸为{3，4，5}的三个窗口，分别异步挂载两个卷积核，进行卷积操作。此外，还需要使用 ReLU 激活函数处理卷积结果，使得输出的特征图只保留正数部分，从而增强模型的非线性能力。实现方法如下：

```
conv_out = F.relu(conv(embed_sents))
```

3. 池化层

池化层接收卷积层输出的特征图后，采用最大池化方法降采样特征图中的典型特征，即最大特征值。并使用分窗口的方式，将这些典型特征值组合成一个新的池化特征向量，且向量长度等于每个窗口设定的卷积核数。对于卷积层输出的 conv_out 结果，采用如下方法实现池化处理：

```
sent_features.append(F.max_pool1d(conv_out, conv_out.size(2)))
```

4. 全连接层

全连接层接受经卷积及池化处理得到的一组特征向量，通过加权及线性变换处理后，将这些特征向量集成映射为一组定长的一维特征向量，以便于后续外接 sigmoid 函数输出分类结果。实现方式如下：

```
logits = self.fc(sent_features)
```

7.4 数据加载

本环节主要使用 torch.utils.data 模块的 DataLoader 方法，完成数据加载工作。在神经网络模型的训练过程中，需要将数据集划分为若干个 batch 进行训练，从而优化模型收敛所需要的时间。实现方式如下：

```
DataLoader(
    dataset=DataLoad("train"),
    batch_size=64,
    collate_fn=partial(collate_fn_dl, word2id)
)
```

其中，首先需要使用自定义的数据容器类 DataLoad，将评论文本训练集转化为[(label,sentment),(),...]类型的输入序列。然后，依赖于预加载的词向量文件 Vocal 对象，将所有输入序列中的 sentment 对象的字词，转化为 word2id 中对应的 id，实现文本的数值化表示。

7.5 模型训练及评估

本环节使用经过数据预处理操作得到的训练集及测试集，依赖预先定义的 CNN_Classifier，结合随机梯度下降方法 Adam()、学习率自适应方法 ReduceLROnPlateau()及二元交叉熵（BCE）损失函数 nn.BCELoss()，完成模型的迭代训练与预测。此外，在模型应用的过程中，选用精确率作为度量指标，在控制台实时反馈每次迭代周期的模型泛化性能。

7.5.1 DeepModel 初始化

在自定义 DeepModel 类中的初始化方法__init__()中：

首先，需要初始化定义：迭代次数、学习率、学习率调度器参数、滤波器数量及窗口大小等参数，如表 7-1 所示。

表 7-1 网络模型初始化参数列表

特征图列维度（emb_size）	300
卷积核数量（num_filters）	2
窗口尺寸（window_sizes）	[3,4,5]
学习率（learning_rate）	0.0015
迭代周期（epoches）	8
学习率下降比例（factor）	0.3
等待轮次（patience）	1

然后，还需根据自适应学习率优化算法 Adam 初始化优化器 optimizer，并指定所需更新的超参列表，避免在无需学习的参数上浪费时间。实现方法如下：

```
self.optimizer = optim.Adam(self.model.parameters(), lr=self.learning_rate)
```

此后，用 ReduceLROnPlateau()方法，定义一个学习率调度器，主要用于在训练过程中自动控制 Adam 优化算法中的学习率大小，提高模型的收敛效率。实现方法如下：

```
ReduceLROnPlateau(self.optimizer,factor=self.lr_decay,patience=self.patience)
```

最后，还需要定义一个二元交叉熵损失函数 nn.BCELoss()，计算 TextCNN 模型预测输出与目标值之间的损失 loss。具体代码如下：

```
self.loss_fn = nn.BCELoss()
```

7.5.2 训练评估方法

在 DeepModel 类中还需要定义 train_and_eval 方法，主要用于对 TextCNN 模型进行训练和测试。模块代码逻辑流程如下：

1. 以迭代周期 epoches 作为循环变量，遍历 train_loader 中的所有批次数据，并用当前批次的数据参数来训练分类模型。

2. 在每一轮迭代周期中，预先清空优化器的梯度信息，以便于后续正、反向传播计算时，重新计算新的参数梯度。主要代码如下：

```
for labels, sentences, lengths in train_loader:
    self.model.train()
    self.optimizer.zero_grad()
```

3. 使用 torch.sigmoid()函数将计算结果映射到[0,1]上，得出文本的类别概率值，从

而实现文本的情感二分类。实现方式如下：

```
torch.sigmoid(self.model(sentences)).squeeze(1)
```

4. 定义二元交叉熵损失函数 BCELoss()，计算神经网络的预测结果与实际分类标签之间的差异损失 loss。并依赖 loss 通过梯度下降法，完成可学习参数的迭代学习。主要代码如下：

```
loss = self.loss_fn(probs, labels.float())
losses += loss.item()
self.optimizer.step()
```

5. 使用自定义的模型性能度量方法 test()，采用准确率作为性能度量指标，分迭代周期对模型在测试集上的泛化性能进行评估，并将模型准确率及平均损失打印到控制台中。主要代码如下：

```
if step % self.print_step == 0:
    print("Epoch {}: {}/{} {:.2f}% finished, Loss: {:.4f}".format(
            e, step,len(train_loader),
            100 * step/len(train_loader),
            losses/self.print_step
    ))
```

7.5.3 性能度量方法

在 DeepModel 类中还需要定义 test()方法，实现了模型在测试数据集上的泛化性能度量过程。该模块的代码逻辑如下：

1. 遍历加载测试集中的每个数据批次 batch，使用 TextCNN 模型计算预测结果，并将输出投影到[0,1]区间上，从而实现预测类别的获取；

```
probs = torch.sigmoid(self.model(sentences)).squeeze(1)
pred_labels = torch.round(probs)
```

2. 计算单个批次 batch 的损失值，并依赖基于批次的平均损失值 loss，通过优化目标函数最小化损失值的方法，实现求解最优权重过程的快速收敛；

```
loss = self.loss_fn(probs, labels.float())
losses += loss.item()
```

3. 通过预测结果与实际类别标签的比对，完成准确率的计算，并适时更新模型的最佳准确率 best_acc；

```
count += len(labels)
correct_num += (pred_labels.long() == labels).sum().item()
acc = correct_num / count
```

4. 在控制台打印所有迭代周期 epoch 的平均准确率及平均损失信息，如图 7-3 所示。

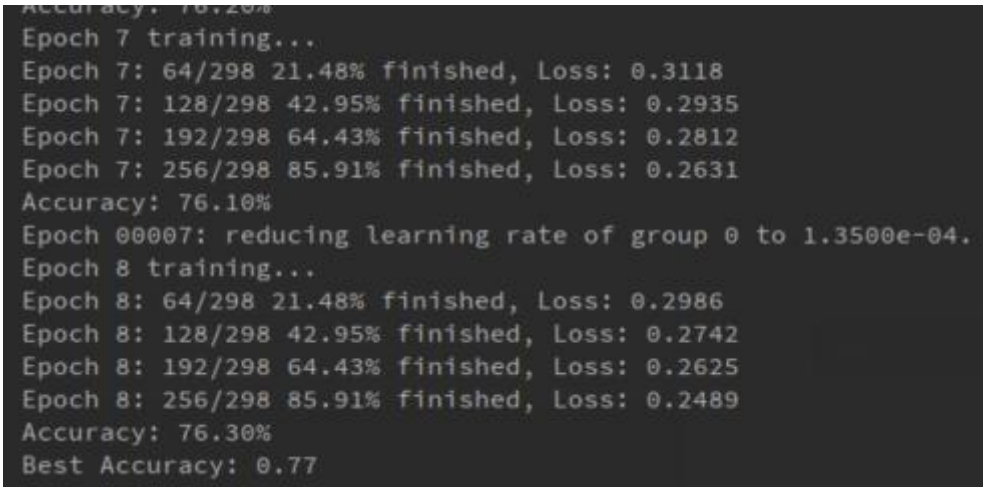

图 7-3 模型预测评估结果（节选）

7.6 实验结果分析

将训练集分批次 batch 作为样本，来迭代性地更新模型的权重的过程中。在每次迭代更新前，都要计算当前批次数据在 TextCNN 模型中预测类别与真实类别标签间的损失值，并通过反向传播方法，更新可学习参数的梯度，从而使模型的泛化性能得到不断地优化。

结合图 7-3 所示的模型预测评估结果日志，以 TextCNN 第八次迭代周期为例，其基于批次的损失值变化如表 7-2 所示。在每次迭代周期中，当模型的损失值逐渐降低时，代表了模型能够更好地拟合训练数据，开始逐渐学习文本的典型特征，表明了模型已经处于可学习参数逐渐趋于最优解的良性状态。

表 7-2 基于批次的损失值变化情况（epoch 8）

	Batch1	Batch2	Batch3	Batch4
Loss	0.2986	0.2742	0.2625	0.2489
Accuracy	76.30%			

同样如图 7-3 所示，在基于 TextCNN 的文本情感分类任务中，模型的最佳准确率（Best Accuracy）为 77%，损失值最终趋于 0.2489。从实际效果来看，TextCNN 算法在处理中文情感分类任务时，泛化性能强于朴素贝叶斯、K 近邻及 K 均值算法，弱于随机森林、支持向量机及逻辑回归算法。其原因可能是：

1. 全局信息学习能力欠佳

TextCNN 算法虽然能通过 N-gram 方法，充分提取字词间的局部关联性信息。但是也容易发生过拟合现象，常常忽略句子级别的全局语义信息。

2. 数据集规模的限制

相较于机器学习算法，TextCNN 在建立网络模型的过程中，需要投入更多的数据用于训练，才能获得更好的文本情感分类效果。针对于本次实践中的两万余条网易云音乐歌单评论数据，在数据质量不高的前提下，无法充分发挥 TextCNN 模型的能力。

3. 模型参数的复杂性

传统机器学习算法由于其可调整参数较少，一定程度上减少了调参效果不佳导致模型泛化性能减弱情况的发生。而在 TextCNN 算法中包含许多可学习参数，正确的设置这些参数，对于 CNN 的表现至关重要。

总而言之，基于 TextCNN 的单一神经网络模型，在处理中文评论文本情感二分类任务时，能够通过 N-gram 方法充分学习字、词及词组级别的局部语义信息，从而获得较好的文本情感分类效果。然而，相较于逻辑回归等有监督机器学习方法，此类方法不擅长处理小数据集及异常样本等情况，还会太过关注局部语义信息，从而导致过拟合现象的发生。综合来看，在基于网易云音乐歌单中文评论的基础上，展开文本情感二分类任务时，推荐使用具有模型解释性强及容错性高等特性的有监督机器学习模型，比如逻辑回归及支持向量机等。

第 8 章　总结与展望

本文采用基于情感词典、无监督分类算法、情感词典与有监督分类算法融合及 TextCNN 算法，针对网易云歌单中文评论文本，依次展开了中文情感识别研究工作。通过实践上述四种关于文本情感识别研究领域的主流实用方法，依赖准确率作为模型性能度量指标，简洁明了地体现了不同分类算法在处理中文评论文本情感极性分类问题时的泛化性能。通过本次实践验证可知，虽然机器学习方法弥补了情感词典不擅长处理未定义情感词的缺陷，但是此类方法需在有监督情景下才能保证分类性能，无监督聚类算法不擅长处理文本情感分类问题。此外，虽然 TextCNN 能够更好地学习评论文本的上下文语义信息，但是此类单一神经网络模型在处理小数据集及异常数据时，实际分类效果略差于逻辑辑回归或支持向量机算法搭建的文本情感分类模型。

在未来，学生希望利用深度学习方法，针对于文本情感极性分析问题，展开更为深入的研究。此外，本次实践过程中大量使用了专门用于人工智能领域开发的 Python 第三方工具包，并没有锻炼在算法层面上的独立编程开发能力。因此在未来，学生希望更深入的学习深度学习算法及建模流程中所涉及到的技术原理，尝试更多地独立编程实现相关模型工具。并通过学习当下更为前沿的人工智能技术理论，在深度学习算法层面上进行更多地尝试，致力于在人工智能专业领域有更为深入的发展。

致　谢

时光匆匆，岁月荏苒。眨眼间，四载大学生活即将告一段落。回首往事，经历酸甜苦辣，铭刻于心，难以言表。在此，谨向所有给予我支持与帮助的老师、家人及同窗们表达最诚挚的感激之情。并在此祝愿所有的朋友们，身体健康、万事如意，都能够实现内心期许。

吴泞宇

2023 年 5 月于长春

参考文献

[1] 王海宁.自然语言处理技术发展[J].中兴通讯技术,2022,28(02):59-64.
[2] 王颖洁,朱久祺,汪祖民,白凤波,弓箭.自然语言处理在文本情感分析领域应用综述[J].计算机应用,2022,42(04):1011-1020.
[3] 王婷,杨文忠.文本情感分析方法研究综述[J].计算机工程与应用,2021,57(12):11-24.
[4] 杨立公,朱俭,汤世平.文本情感分析综述[J].计算机应用,2013,33(06):1574-1578+1607.
[5] 魏嵬,孙雪松,李林峰等.基于文本的情感分析方法论述[J].数字技术与应用,2022,40(12):1-3+22.DOI:10.19695/j.cnki.cn12-1369.2022.12.01.
[6] 刘慧慧,王爱银,刘禹彤.基于SVM的文本情感分析——以新冠疫情事件为例[J].信息技术与信息化,2023,No.274(01):37-40.
[7] 刘丹丹,邱恒清.基于SVM的中文微博情感识别与分类研究[J].中国新通信,2015,17(21):48-51.
[8] 李首政,王琪.基于支持向量机的微博情感分析方法研究[J].现代计算机,2022,28(19):63-66+80.
[9] 刘祉燊,张倩,周菠等.基于支持向量机的中文文本情感分析方法研究[J].科技创新与应用,2022,12(32):27-30.DOI:10.19981/j.CN23-1581/G3.2022.32.007.
[10] 王志颖.基于Spark和机器学习的文本情感分析研究[D].华中师范大学,2021.
[11] 季立堃.基于深度学习的文本情感分析技术研究[D].北京邮电大学,2019.
[12] 吴蕙羽.基于Python技术分析小说人物关系和社交网络[J].电脑编程技巧与维护,2020,(06):61-63.
[13] 杜茂康,李晓光,刘崇.融合遗传算法的特定领域情感词库构建[J].重庆邮电大学学报(自然科学版),2022,34(04):576-584.
[14] 刘笑.基于社交网络文本的情感分析[J].信息与电脑(理论版),2021,33(22):171-175.
[15] 木欣喜.融合情感词典和注意力机制的Bi-LSTM网评文本情感分析研究[D].南昌大学,2022.DOI:10.27232/d.cnki.gnchu.2022.002735.
[16] Hung Lai Po,Alias Suraya. Beyond Sentiment Analysis: A Review of Recent Trends in Text Based Sentiment Analysis and Emotion Detection[J]. jaciii,2023,27(1).
[17] Yao Lisha. Sentiment analysis based on CNN-LSTM hotel reviews[J]. Journal of Physics: Conference Series,2022,2330(1).
[18] Jyostna Devi Bodapati, N. Veeranjaneyulu, Shareef Shaik. Sentiment Analysis from Movie Reviews Using LSTMs[J]. Ingénierie des Systèmes d'Information,2019,24(1).