毕业论文

|  |  |
| --- | --- |
| 题目： | 多源异构关联大数据采集和分析平台的设计 |
|  |  |

姓 名： XXXX

学 号： XXXXX

院 系： XXXXX

专 业： XXXXX

指导教师： XXXXX

二〇二二年X月XXX日

摘要

随着大数据时代的到来，网络信息传播迅速，覆盖范围广，开源信息大量聚集，为训练人工智能模型奠定了良好的基础。当前网络数据来源广泛，比如新闻微博数据、Github的开源代码数据、以及社交网络Twitter的评价数据等。如何从大量的开源数据中爬取、筛选有价值的内容，进行数据分析，并服务于人工智能技术是当前的研究难题。

近些年来，深度学习技术利用海量的数据和大型神经网络来从数据中自动学习和挖掘特征信息，在图像识别、自然语言处理、语音识别、机器人自动化等领域均有重大突破。在金融领域，期刊和新闻等文本数量快速迭代积累，数据规模不断丰富和扩大，因此可以用深度学习技术从文本中评论的倾向性，进而可用于分析和判断未来走势。

本文研究多源异构关联大数据采集和分析问题，通过scrapy和hadoop平台采集多源异构文本数据，并用数据挖掘方法进行数据分析可视化，同时采用深度学习技术训练情感分析模型，最后将数据和模型部署在web平台和andrioid app，支持数据统计、情感分析、专题检索等功能。

**关键词：**开源数据采集；深度学习；数据挖掘

Design of Multi source Heterogeneous Correlation Big Data Acquisition and Analysis Platform

Abstract

With the arrival of the big data era, network information spreads rapidly, covers a wide range and gathers a large number of open-source information, laying a good foundation for training artificial intelligence models. The current network data sources are extensive, such as news micro blog data, Github open-source code data, and social network Twitter evaluation data. How to crawl and filter valuable content from a large number of open-source data, conduct data analysis, and serve AI technology is a current research problem.

In recent years, deep learning technology uses massive data and large neural networks to automatically learn and mine feature information from data. It has made significant breakthroughs in image recognition, natural language processing, speech recognition, robot automation and other fields. In the financial field, the number of periodicals, news and other texts has accumulated rapidly and iteratively, and the data scale has been continuously enriched and expanded. Therefore, in-depth learning technology can be used to comment on the tendency of texts, which can then be used to analyze and judge the future trend.

This paper studies the collection and analysis of multi-source heterogeneous related big data. It collects multi-source heterogeneous text data through the sketch and hadoop platforms, and uses data mining methods for data analysis visualization. At the same time, it uses deep learning technology to train the emotional analysis model. Finally, it deploys the data and model on the web platform and android app to support data statistics, emotional analysis, thematic retrieval and other functions.

**Key words:** Open-source Data Collection; Deep Learning; Data Mining

目录

[摘要 1](#_Toc114998482)

[Abstract 2](#_Toc114998483)

[第1章 绪论 6](#_Toc114998484)

[1.1 研究工作的背景与意义 6](#_Toc114998485)

[1.2 国内外研究现状 7](#_Toc114998486)

[1.2.1 文本情感分析发展 7](#_Toc114998487)

[1.2.2 专题检索发展 9](#_Toc114998488)

[1.3 本文的主要贡献 9](#_Toc114998489)

[1.4 本文组织架构 9](#_Toc114998490)

[第2章 深度学习文本处理技术 10](#_Toc114998491)

[2.1 文本向量化表示方法 10](#_Toc114998492)

[2.1.1 离散表示 10](#_Toc114998493)

[2.1.2 分布表示 12](#_Toc114998494)

[2.2 基于深度学习的文本处理技术 13](#_Toc114998495)

[2.2.1 适用于文本分析的卷积神经网络 13](#_Toc114998496)

[2.2.2 适用于序列处理的循环神经网络 15](#_Toc114998497)

[2.3 本章小结 19](#_Toc114998498)

[第3章 数据收集与预处理 21](#_Toc114998499)

[3.1 多源异构大数据采集 21](#_Toc114998500)

[3.2 开源信息数据库 23](#_Toc114998501)

[3.3 文本分析流程 24](#_Toc114998502)

[3.3.1 文本获取与筛选 24](#_Toc114998503)

[3.4 数据标注及预处理 25](#_Toc114998504)

[3.5 数据统计及可视化 26](#_Toc114998505)

[3.6 本章小结 30](#_Toc114998506)

[第4章 基于深度学习的文本分析 31](#_Toc114998507)

[4.1 多模型情感分类器训练 31](#_Toc114998508)

[4.2 数据集与分类模型性能分析 34](#_Toc114998509)

[4.3 实验结果对比与分析 35](#_Toc114998510)

[4.4 本章小结 36](#_Toc114998511)

[第5章 系统界面设计 37](#_Toc114998512)

[5.1 UI界面设计及美化 37](#_Toc114998513)

[5.2 手机App设计与系统互联 40](#_Toc114998514)

[第6章 全文总结与展望 43](#_Toc114998515)

[6.1 全文总结 43](#_Toc114998516)

[6.2 后续工作展望 43](#_Toc114998517)

[致谢 44](#_Toc114998518)

[参考文献 45](#_Toc114998519)

第1章 绪论

1.1 研究工作的背景与意义

近几年来，随着互联网的不断普及和人工智能时代的到来，互联网技术已经领先全球，尤其是移动互联网技术的发展更是给人类的日常生活带来了巨大的改变。移动互联网时代的到来伴随着的是原来在PC端使用的一些平台逐渐转化到移动端，比如即时通讯软件、社区软件、支付软件等等，他们在发展的同时更是带来了大数据。随着大数据时代的到来，人们积累数据的意识越来越强，如何从积累的数据中获取我们想要的信息越来越受到关注。社交网络、twitter或facebook上的公众评论、公司电子邮件记录、疾病发展、网站访问等。越来越多的可以连接到网络的数据被保存下来，因此学习分析这些数据可以帮助我们找到信息我们需要在凌乱的连接关系中。

过去十年，社交媒体和即时通讯软件一直是人们常用的互联网工具，尤其随着大数据时代的到来，在社交软件上产生的数据每天都在大幅度增长，比如微博平台，每天都在产生新的大量数据，人们常用其发表自己对新鲜事物的看法和评论。尤其是在财经领域，投资者们需要时刻关注股市等金融产品的信息，进而合理投资，排除掉冗余因素的影响，以达到使得个人资产最大化的目的。因此各大财经投资网站得以迅速发展，实时地播报与财经市场发展相关的消息及评论信息，例如国家地方的经济政策、公司财报等。由于股票价格与多种因素相关，比如国家的经济发展、公司的实际运营情况、以及法律法规的制定等，这些因素将导致股票价格存在高度的波动性与不确定性，成为当前金融界的一大难题。

然而互联网信息量往往是爆炸式增长的，特别是财经文木具有更新速度快、

数量大等特点，选取新浪财经网站对其发布的新闻数量进行初步统计，财经新闻文木量呈现出了逐年上涨的趋势。单靠人工方式收集文木信息并逐一进行阅读、分析、整理等一系列操作成木极高、耗时极长，很容易造成重要信息的遗漏，然而财经领域对于文木信息的时效性要求极高，完全依赖人工的处理方式明显是不可取的。

如何对这些网络平台发布出的文本进行处理，并从中提取出所表现出来的情感倾向对于数据分析具有重要研究意义。

近年来，深度学习技术利用海量数据和大型神经网络来自动化地从数据内部挖掘有用特征信息，在图像识别、自然语言处理、语音识别等众多领域均取得了重大突破。深度学习技术在时序数据分析和预测上也取得了众多研究成果，比如针对股票市场的数据分析和预测，华泰证券金融工程研究所使用BP神经网络对A股股票价格进行预测和分析；广发证券量化交易团队使用深度神经网络模型，对A股股指期货高频价格进行预测等。在自然语言处理领域，深度学习技术取得的成果甚至可以和人类相提并论，比如GPT3在散文创作、诗歌撰写可以达到以假乱真的效果。Bert模型在完形填空，阅读理解等任务上几乎达到了和人类一样的水平。

1.2 国内外研究现状

1.2.1 文本情感分析发展

文本情感分析(Sentiment Analysis)是文木分类问题中的一个重要研究方向，

属于深度学习自然语言处理(Natural Language Processing, NLP)领域，主要是将

文本针对其描述对象所表现出来的情感倾向性进行分类，识别出文木背后隐含的

观点是积极的还是消极的，进而扩大研究范围得到社会公众对事物所表现出来的

情感倾向。情感分析问题从文木处理的粒度出发，可以分为两种—粗粒度情感

分析与细粒度情感分析。情感分析（也称为意见挖掘或情感AI）是利用自然语言处理、文本分析、计算语言学和生物特征来系统地识别、提取、量化和研究情感状态和主观信息。情绪分析广泛应用于客户材料的声音，如评论和调查回复、在线和社交媒体，以及医疗保健材料，应用范围从营销到客户服务到临床医学。随着深入语言模型（如RoBERTa）的兴起，也可以分析更困难的数据域，例如，作者通常不太明确地表达其观点/情感的新闻文本。情绪分析至关重要，因为它有助于深入了解客户对您品牌的看法。客户反馈，无论是通过社交媒体、网站、与服务代理的对话还是任何其他来源，都包含了宝贵的有用商业信息，但仅仅知道客户在谈论什么是不够的。了解他们的感受会让你最深入地了解他们的经历。情绪分析是理解这些经历的一种方法。有时被称为“意见挖掘”，情绪分析可以让用户知道公众对业务的任何方面的意见是否发生了变化。如果用户想改进产品、培训销售代表或客户服务代理，或创建新的营销活动，情绪得分的峰值或低谷会给用户一个起点。

粗粒度情感分析方法以段落、语句为主要研究对象，可以直接判断出文章整体表现的情感倾向性。但是粗粒度分析的功能却仅限于此，它无法识别出文木是对哪项事物表现出了怎样的情感倾向。然而文木中往往存在转折关系，针对整体的情感判决无法表征出对文中提及的多个实体的情感倾向。细粒度情感分析从字词层面出发，对文木所提及的实体以及与其相关联的情感开展分析，处理方法更为细致。相较于文木整体，细粒度情感分析更关注于文木内部的特征，通常包含匹配情感词、评价对象识别、情感倾向分析等处理任务，文本处理方式也相对复杂。常用的文本情感分析方法主要有基于统计的方法和基于深度学习的方法。

情感分析是文本挖掘的大伞中的一部分，也称为文本分析。这种类型的分析从许多文本来源中提取意义，例如调查、评论、公共社交媒体，甚至网络上的文章。然后根据文本中表达的情感给每个子句分配一个分数。例如，-1表示消极情绪，+1表示积极情绪。这是使用自然语言处理（NLP）完成的。

基于统计的方法主要基于情感词典展开情感分析工作，通过字符串匹配查找情感词的方式进行文本分析，其核心在于情感倾向词典的构建，往往采用人工筛选的方式，将具有明显情感倾向的词语收集起来制作情感词典，词典的质量直接关系到文本情感分析结果。文献[]均采用情感词典方法进行文木情感分析。

在情感分析问题的研究历程中针对情感词典的构建方式展开了大量研究，制作了如知网(HowNet)情感词典、台湾大学(NTSUSD)简体中文情感极性词典等通用的情感词典;提出了在已有情感词典上进行扩展的方法，比如通过统计词语关联情感倾向性的概率进行情感词的发掘与扩充;使用同义词对初始情感词典进行扩展;使用词向量之间的空间相似性扩展词典等方法。

随着深度学习的最新进展，算法分析文本的能力已经大大提高。创造性地使用先进的人工智能技术可以成为进行深入研究的有效工具。我们认为，重要的是要根据以下内容对即将到来的客户关于品牌的谈话进行分类：

基于深度学习的方法的基本思想是使用大量样本训练各种神经网络模型来发现学习文木与情感倾向的潜在联系，通过构建端到端的学习模式，将大部分的特征提取任务黑箱化，尽可能省略人工参与的步骤。众多研究人员以卷积神经网络模型、循环神经网络模型、注意力机制为基础，开展文木分析处理领域的研究逐步提出了序列到序列、Transformer, GPT, BERT等优秀的文木分析模型，并在此基础上进行了更为深人的研究，这些新兴的深度学习模型在情感分析任务上表现出了良好的性能，为情感倾向分析提供了更多的选择。

1.2.2 专题检索发展

检索网络对于搜索和索引是必不可少的。深度学习利用各种排名损失来学习一个对象的嵌入，来自同一类的对象的嵌入比来自不同类的对象的嵌入更接近。本文比较了各种著名的排名损失的公式和应用。深度学习的检索正式的说法为度量学习(ML)。在这个学习范式中，神经网络学习一个嵌 入—— 比如一个128维的向量。这样的嵌入量化了不同对象之间的相似性，如下图所示。学习后的嵌入可以进行搜索、最近邻检索、索引等，使用用排序损失训练的深度网络，使搜索和索引成为可能。其中，对比损失、三元组损失等为用深度学习技术进行检索的常见损失函数。

1.3 本文的主要贡献

本文研究多源异构关联大数据采集和分析问题，通过scrapy和hadoop平台采集多源异构文本数据，并用数据挖掘方法进行数据分析可视化，同时采用深度学习技术训练情感分析模型，最后将数据和模型部署在web平台和andrioid app，支持数据统计、情感分析、专题检索等功能。

1.4 本文组织架构

第1章 绪论：介绍了开源信息数据采集问题的背景，意义和国内外发展现状，及本文使用大数据平台进行采集和数据挖掘技术进行分析的技术路线；

第2章 深度学习模型背景知识介绍：介绍了基于深度学习的文本处理技术所使用的自然语言处理经典模型架构及训练方法；

第3章 数据收集与预处理方法介绍：介绍了本文验证提出方法的环境搭建，包括数据集的获取及预处理，爬虫和分布式平台的部署；

第4章 基于深度学习的文本分析方法介绍：介绍了本文使用的深度学习模型，在其上做出的改进，训练超参数的设置，衡量指标及结果的展现；

第5章 web端和andriod端的展示：介绍了如何在网页端和移动端部署开源信息采集系统和情感分析及数据检索模型；

第6章 总结：介绍了本文的主要工作，存在的不足及未来值得改进的方向。

第2章 深度学习文本处理技术

2.1 文本向量化表示方法

文本表示的意思是把字词处理成向量或矩阵，以便计算机能进行处理。文本表示是自然语言处理的开始环节。文本表示按照细粒度划分，一般可分为字级别、词语级别和句子级别的文本表示。字级别（char level）的如把“邓紫棋实在太可爱了，我想养一只”这句话拆成一个个的字：｛邓，紫，棋，实，在，太，可，爱，了，我，想，养，一，只｝，然后把每个字用一个向量表示，那么这句话就转化为了由14个向量组成的矩阵。

文本表示分为离散表示和分布式表示。离散表示的代表就是词袋模型，one-hot（也叫独热编码）、TF-IDF、n-gram都可以看作是词袋模型。分布式表示也叫做词嵌入（word embedding），经典模型是word2vec，还包括后来的Glove、ELMO、GPT和BERT。

2.1.1 离散表示

关于文本的离散表示，首先介绍词袋模型。给定1000篇新闻文档，把这些文档拆成一个个的字，去重后得到3000个字，然后把这3000个字作为字典，进行文本表示的模型，叫做词袋模型。这种模型的特点是字典中的字没有特定的顺序，句子的总体结构也被舍弃了。词袋模型通常包含one-hot、TF-IDF和n-gram等文本表示方法。

首先介绍one-hot模型，比如有两句话“邓紫棋太可爱了，我爱邓紫棋”，“我要看邓紫棋的演唱会”，把这两句话拆成一个个的字，整理得到14个不重复的字，这14个字决定了在文本表示时向量的长度为14。

要对两句话进行数值表示，那么先构造一个2×14的零矩阵，然后找到第一句话中每个字在字典中出现的位置，把该位置的0替换为1，第二句话也这样处理。只管字出现了没有（出现了就填入1，不然就是0），而不管这个字在句子中出现了几次。One-hot表示方式主要存在以下几点问题：

第一个问题是数据稀疏和维度灾难。数据稀疏也就是向量的大部分元素为0，如果词袋中的字词达数百万个，那么由每篇文档转换成的向量的维度是数百万维，由于每篇文档去重后字数较少，因此向量中大部分的元素是0。而且对数百万维的向量进行计算是非常耗费资源的。

第二个问题是没有考虑句中字的顺序性，假定字之间相互独立。这意味着意思不同的句子可能得到一样的向量。比如“我太可爱了，邓紫棋爱我”，“邓紫棋要看我的演唱会”，得到的one-hot编码和上面两句话的是一样的。

第三个问题是没有考虑字的相对重要性。这种表示只管字出现没有，而不管出现的频率，但显然一个字出现的次数越多，一般而言越重要（除了一些没有实际意义的停用词）。

然后介绍TF-IDF模型，TF-IDF用来评估字词对于文档集合中某一篇文档的重要程度。字词的重要性与它在某篇文档中出现的次数成正比，与它在所有文档中出现的次数成反比。TF是词频的意思，用来衡量字在一篇文档中的重要性，计算公式为某文档中某字出现的次数除以该文档的总字数。IDF表示逆文档频率，衡量某个字在所有文档集合中的常见程度。当包含某个字的文档的篇数越多时，这个字的重要性越低。计算公式为全部文档的数量除以包含某字的文档的篇数，最后得到的TF-IDF的值为TF和IDF的乘积，其思想比较简单，但是非常实用，然而TF-IDF方法仍然存在数据稀疏的问题，也没有考虑字的前后信息。

最后介绍n-gram模型，上面词袋模型的两种表示方法假设字与字之间是相互独立的，没有考虑它们之间的顺序。于是引入n-gram（n元语法）的概念。n-gram是从一个句子中提取n个连续的字的集合，可以获取到字的前后信息。一般2-gram或者3-gram比较常见。比如“邓紫棋太可爱了，我爱邓紫棋”，“我要看邓紫棋的演唱会”这两个句子，分解为2-gram词汇表：｛邓，邓紫，紫，紫棋，棋，棋太，太，太可，可，可爱，爱，爱了，了，了我，我，我爱，爱邓，我要，要，要看，看邓，棋的，的，的演，演，演唱，唱会，会｝于是原来只有14个字的1-gram字典（就是逐字进行划分的方法）就成了28个元素的2-gram词汇表，词表的维度增加了一倍。N-gram表示方法的好处是可以获得更加丰富的上下文特征，提取字的上下文信息，考虑了字之间的顺序性。其问题是没有解决数据稀疏和词表维度过高的问题，并且词表维度会随着n的增大而变高。

因此，文本的离散化表示存在着数据稀疏、向量维度过高、字词之间的关系无法度量的问题，通常适用于浅层的机器学习模型，不适用于拟合复杂数据的深度学习模型。

2.1.2 分布表示

文本向量的分布表示使用了共现矩阵结合SVD降维处理方法，实现了使用上下文表示单词，共现是指不同的词同时出现的文档数，对于其存在的维度灾难和数据稀疏的问题，应该对高维信息进行降维，因此需要通过降维的方法来解决，一种常用的方法是奇异值分解。

第二种方案是word2vec方法，它是一种高效训练词向量的模型，其基本出发点和分布式表示比较类似，即上下文相似的两个词，他们的词向量表示应该也相似，比如香蕉和梨在句子中可能经常出现在相同的上下文中，因此这两个词的表示向量应该就比较相似。Word2vec的基本思想是句子之中相近的词之间是有联系的，比如今天后面经常出现上午、下午。所以它的基本思想就是用词来预测词。（准确的说，word2vec仍然是一种编码方式，将一个个的词给编码成向量，但是被他编码而成的向量并不是随便生成的，而是能够体现这些单词之间的关系（如相似性等）），word2vec主要包含两个模型，第一个是skip-gram模型，即用当前词来预测上下文，相当于给定一个词去预测前面和后面的词汇。第二个是连续词袋模型，即通过上下文来预测当前值，相当于从一句话种扣掉一个词，然后猜这个词的语义。

Skip-gram模型的实现方式为，在每一次迭代中都取一个词作为中心词汇，尝试去预测它一定范围内的上下文词汇。所以这个模型定义了一个概率分布：给定一个中心词，某个单词在它上下文中出现的概率。我们会选取词汇的向量表示，从而让概率分布值最大化。重要的是，这个模型对于一个词汇，有且只有一个概率分布，这个概率分布就是输出，也就是出现在中心词周围上下词的一个输出。拿到一个文本，遍历文本中所有的位置，对于文本中的每个位置，我们都会定义一个围绕中心词汇大小为2m的窗口，这样就得到了一个概率分布，可以根据中心词汇给出其上下文词汇出现的概率。​ 现在这个已知的句子就是我们的一个样本，我们要进行第一次迭代，在迭代的过程中，我们的损失函数需要这里表示在给定中心词的情况下，在2m窗口内的所有其他词出现的概率（T表示词库里所有词的总数）。目标是要通过调节参数，从而最大化这个函数（因为这个函数越大，表示与实际情况越吻合）。另外，根据平时的习惯，我们通常喜欢最小化损失函数，而不是最大化损失函数。因此我们对该函数取负对数，且除以T，得到新的损失函数（对数似然函数）。

连续词袋模型与跳字模型类似，最大的不同在于连续词袋模型假设基于某中心词在文本序列前后的背景词来生成该中心词。例如：‘我’，‘爱’，‘红色’，‘这片’，‘土地’，窗口大小为2，就是用‘我’，‘爱’，‘这片’，‘土地’这四个背景词，来预测生成 ‘红色’ 这个中心词的条件概率，即：P（红色|我，爱，这片，土地）给定一个长度为T的文本序列，设时间步t的词为W(t),背景窗口大小为m。则连续词袋模型的目标函数（损失函数）是由背景词生成任一中心词的概率。

2.2 基于深度学习的文本处理技术

2.2.1 适用于文本分析的卷积神经网络

卷积神经网络本质上来说就是几个卷积层加上非线性函数，像ReLu或者tanh。在传统的神经网络中，我们将每个输入神经元与下一层连接。这个叫做全连接层或者仿射层。在CNN中，通常用输入层上的卷积来计算输出。这就导致了局部连接，其中每个输入层的区域都是输出层的神经元连接在一起的。每层都使用了filter，通常会有成千上百的类似于上面有小框框的图那样的操作。还有一些东西叫做pooling(subsampling)层。在训练的过程中，一个CNN会基于任务自动地学习filters的值。例如，在图像分类中，一个CNN可以学会检测边，然后用边在第二层中去检测简单的形状，比如在高层中面部形状。最后一层就是一个使用这些高等特征的分类器。这里的计算有两方面值得注意：位置不变和合成(compositionality)。每个filter把局部的低维特征合成为高维特征。这就是CNN在计算机视觉任务里表现良好的原因。与图像像素不同，对于绝大多数自然语言处理的任务，输入都是矩阵形式的句子和文件。矩阵的每一行都对于这一个单词，但是也可能是一个字符。也就是，每一行都是一个表示一个单词的向量。通常来说，这些向量都是word embedding（低维表示），比如word2vec或者GloVe，但是他们也可能是one-hot向量（对应于一个词典）。在视觉中，filter经过图像的每一小块，但是在NLP中，通常使用filter滑过整个矩阵的整个行(words)。因此，filter的宽度通常是与输入矩阵的宽度相同。高度，或者说区域大小是可以变动的，但是25个单词上的滑动框在这个时候是一定的。把上面所有的都放在一起，一个NLP下的CNN如下图所示。

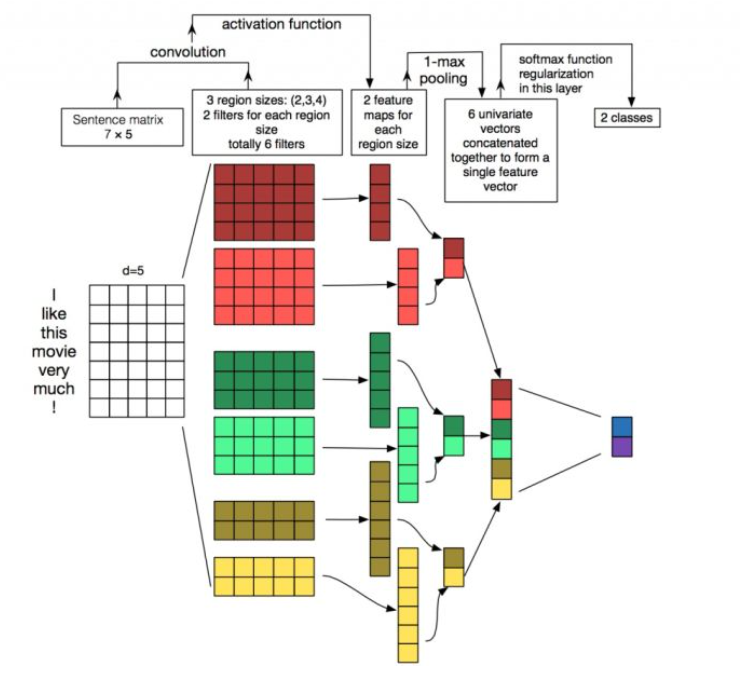


图 2-1 CNN用于处理自然语言处理分类人物的示意图

局部不变性以及局部合成是的对于图像有直观的感觉，但是对于NLP并没有这样很强的直觉。可能很关心句子中哪里会有一个单词出现。像素彼此很靠近的时候可能会有相关的语义，但是相同通常对于单词来说不是一回事。在很多语言中，部分短语可以被分割几个其他的单词。合成方面也不是那么明显。但是，单词很明显都是用同样的方式组合的，比如形容词修饰名词，单数这个，在高维表示中的意义并没有在计算机视觉中的大。给CNN最适合的任务就是分类，比如语义分析或者主题聚类. 卷积和池化算子会丢失一些局部的位置信息，从而使得句子标签变成了Part of Speech标签或者实体提取变得更加困难对于纯粹的CNN框架。

[]从头开始训练CNN，没有使用预先训练好的，像Word2Vec，GloVe。它直接把卷积运用到one-hot向量上。这个作则也提出了一个空间有效的bag-of-words-like的表示来表示输入数据。从而减少神经网络中需要学习的参数。在[5]中，作者使用另外一种非监督式的"region embedding"，也就是通过使用CNN预测文本中的内容来学习。这些论文中的方法似乎对于长文本都是有效的，但是他们对于短文本的表现(像tweets)并不是很好。直觉上来看，这是有道理的，因为使用预先训练好的word embedding来对短文本进行建模将会产生更多的误差，相对于长文本来说。

2.2.2 适用于序列处理的循环神经网络

在自然语言处理任务中，除了文本语义分类任务外，还有生成类的任务，比如机器翻译，文本生成等。这类模型需要充分挖掘自然语言的特性，将源文本充分建模，并输出全局的上下文表示，每个解码的单词都是由输入文本的全局特征加以注意力机制得到的。因此在处理此类任务的文本时，往往需要用到循环神经网络，比如RNN、LSTM等。

传统的神经网络DNN或者CNN网络他们的输入和输出都是独立的。对于这些模型输入的数据跟输出的数据大多是关联不太紧密的场景，但是有些场景输入的数据对后面输入的数据是有关系的，或者说后面的数据跟前面的数据是有关联的。例如，对于文本类的数据，当输入某句话的时候，刚开始输入第一个字的时候，再输入这句话的第二个字时候，其实第二个字要输入什么字其实是跟第一个字是有关联的。所以，对于这样一类的场景，通常是要考虑前面的信息的，以至于引入RNN模型。对于RNN模型为解决这类问题引入了“记忆”这一概念。循环神经网络的循环来源于其每个元素中都执行相同的任务，但是输出依赖于输入和“记忆”两个部分。

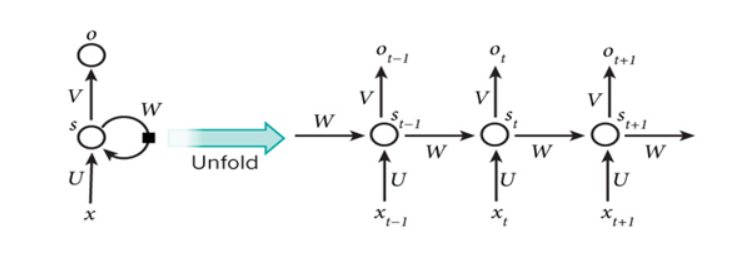


图2-2 RNN网络模型架构

对于循环神经网络，可以把隐状态视作为“记忆体”，捕捉之前时间点上的信息。输出有当前时间及之前所有“记忆”共同计算得到的。但由于是一个有限的矩阵，对于之前的信息并不能完全捕捉到，也会随着时间的变长，对于之前的“记忆”也会“变淡”。对于RNN不同于DNN与CNN，这里的RNN其实整个神经网络都在共享一组参数（U,V,W），这样极大的减小了需要训练的参数。图中的Ot再由写任务下是不存在的，只需要对最后的结果输出就可以。RNN的变种模型还有双向RNN。

双向RNN，通过以上经典的RNN模型，它是只关心当前的输入和之前的“记忆”的，但有些情况下，当前的输入不知依赖于之前的序列元素，还依赖于后面序列的元素。比如，一篇文章，当读第一段时候我们并不知道文章的主体要讲什么内容，但当我们读完第一段的时候需要判断文章主要讲什么内容，这时候就需要读后面的内容才能知道这个文章主要讲的是什么。对于这样的场景需要后面的数据才能更好的预测当前的状态，所以引入了双向RNN，就是为了解决这一类问题的。双向RNN是考虑到了前后的“记忆”，能够更好的关联到前后的信息。

LSTM模型，架构如图2-3所示，是Long Short Term Memory的缩写，也是循环神经网络的一种变体，它致力于解决循环神经网络（RNN）的梯度爆炸或梯度消失问题。LSTM主要致力于解决时间序列预测问题，由遗忘门、输入门、输出门等组成，在众多序列预测任务上取得了良好效果，比如自然语言生成、文本分类任务等。LSTM致力于通过给定的时间序列数据去预测下一状态的结果，这主要通过门控单元和记忆单元实现。每个单元存储最近经历过的状态，每个时间戳的数据都输入进记忆单元中，最终的输出结果通过结合当前记忆单元的状态和上一时刻累积的状态和共同决定。LSTM是被用来设计解决长期依赖问题，有效地缓解了RNN中如果依赖过长导致的梯度消失或梯度爆炸的问题。

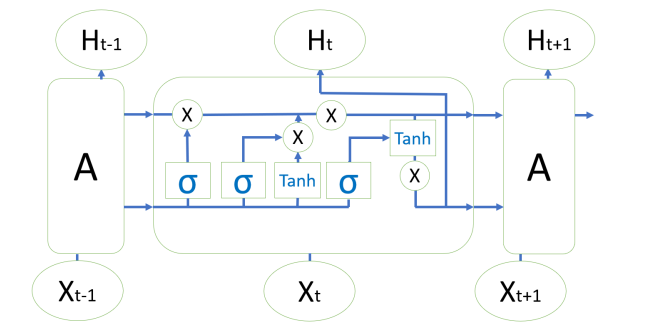


图2-3 LSTM模型架构示意图

Bi-LSTM是Bi-directional Long Short-Term Memory的缩写，其架构如图2-4所示，是由前向LSTM与后向LSTM组合而成。LSTM模型是由在RNN基础上增加了输入门，遗忘门，单元状态，输出门组成。在网络训练过程中，可通过门结构来添加或移除信息，不同神经网络都可通过单元状态上的门结构来决定去记住或遗忘哪些相关信息。

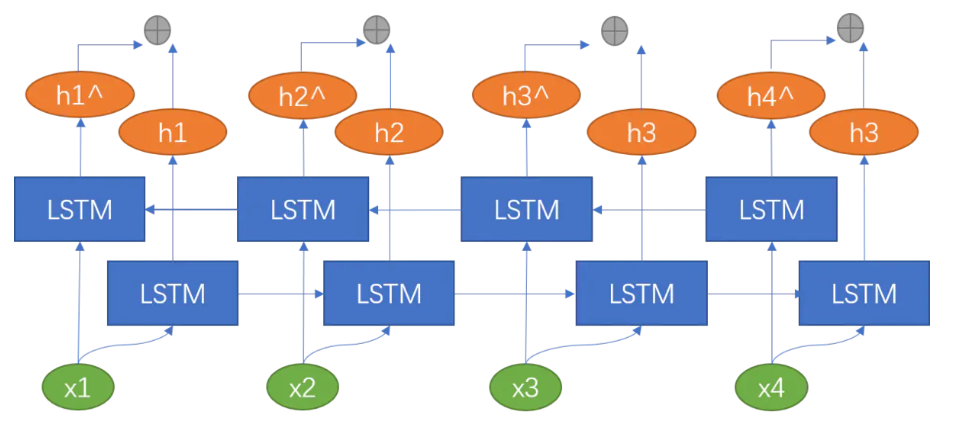


图2-4 BI-LSTM模型架构图

Bi-LSTM模型是由在RNN基础上增加了门机制和记忆单元，有效的防止了梯度爆炸和梯度消失，同时更好的捕捉到较长距离的依赖，使用BI-LSTM可以捕捉到双向的语义依赖。

Transformer是基于多头注意力机制的用于序列处理的神经网络。它也是由经典的编码器-解码器机制实现的神经网络。编码器用于将序列建模成向量化表示，将离散的文本序列映射到连续的状态空间中去。注意力（Attention）机制由Bengio团队与2014年提出并在近年广泛的应用在深度学习中的各个领域，例如在计算机视觉方向用于捕捉图像上的感受野，或者NLP中用于定位关键token或者特征。谷歌团队近期提出的用于生成词向量的BERT[]算法在NLP的11项任务中取得了效果的大幅提升，堪称2018年深度学习领域最振奋人心的消息。而BERT算法的最重要的部分便是本文中提出的Transformer的概念。Transformer中抛弃了传统的CNN和RNN，整个网络结构完全是由Attention机制组成。更准确地讲，Transformer由且仅由self-Attenion和Feed Forward Neural Network组成。一个基于Transformer的可训练的神经网络可以通过堆叠Transformer的形式进行搭建，作者的实验是通过搭建编码器和解码器各6层，总共12层的Encoder-Decoder，并在机器翻译中取得了BLEU值得新高。作者采用Attention机制的原因是考虑到RNN（或者LSTM，GRU等）的计算限制为是顺序的，也就是说RNN相关算法只能从左向右依次计算或者从右向左依次计算，这种机制带来了两个问题：第一是时间片t的计算依赖t-1时刻的计算结果，这样限制了模型的并行能力；第二是顺序计算的过程中信息会丢失，尽管LSTM等门机制的结构一定程度上缓解了长期依赖的问题，但是对于特别长期的依赖现象，LSTM依旧无能为力。Transformer的提出解决了上面两个问题，首先它使用了Attention机制，将序列中的任意两个位置之间的距离是缩小为一个常量；其次它不是类似RNN的顺序结构，因此具有更好的并行性，符合现有的GPU框架。编码器由6个编码block组成，同样解码器是6个解码block组成。与所有的生成模型相同的是，编码器的输出会作为解码器的输入。关于多头注意力机制，其相当于是h个不同的注意力机制的集成，其输出一共分为3步，首先是将数据分别输入到h个自注意力头中，得到h个加权后的特征矩阵，然后将h个特征矩阵按列拼接成一个大的特征矩阵，最后经过一层全连接网络后输出得到编码后的特征向量。与自注意力机制一样，多头注意力机制也引入了short-cut机制。关于编码器-解码器的注意力机制，query值来自于解码器上一时刻的输出，键值和价值来自于编码器的输出，由于在机器翻译中，解码过程是一个顺序的操作过程，也就是当解码第k个特征向量时，只能看到前k-1个解码结果，因此这种情况下的注意力机制叫做遮掩状态下的掩码注意力机制。关于损失层，解码器在输出解码向量后，解码的特征向量经过一层激活函数为softmax的全连接层后，得到反映每个单词概率的输出向量，这时可以通过CTC损失函数训练模型。关于位置编码，Transformer是可以捕捉顺序序列的，也就是说无论句子如何打乱单词顺序，Transformer都会得到类似的结果，比较稳定，且鲁棒性较强，因此Transformer引入了位置编码特征，具体而言，位置编码会在词向量中引入单词的位置信息，这样Transformer即可区分不同位置的单词，常见的位置编码方式有根据数据学习的，自己设计编码规则的。这里通常是使用长度为d的特征向量，以便于对词向量进行单位加的操作。Transformer的设计可以提升性能的关键之处在于将任意两个单词的距离设置为1，这对解决自然语言处理中长期依赖的问题是非常有效的，且Transformer还可以用于计算机视觉领域，比如vision-transformer。Transformer的缺点是其捕捉局部特征的能力较差，Transformer虽然使用了位置编码信息去保留位置信息，但这还是明显不如时许序列处理模型如LSTM处理的过程。

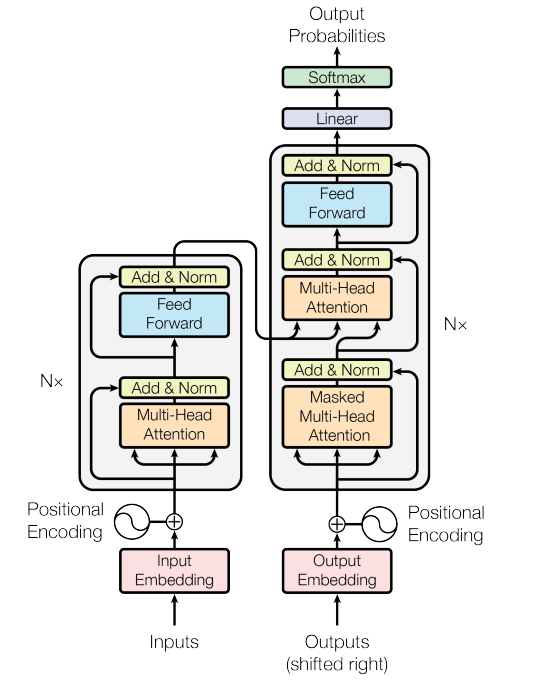


图 2-5 Transformer网络架构图

2.3 本章小结

本章介绍了文中使用的两个深度学习模型，LSTM和BI-LSTM的架构，实现原理及相应的好处，为展开实验部分的介绍奠定了理论基础。

第3章 数据收集与预处理

3.1 多源异构大数据采集

本文的爬虫部分主要是用scrapy和selenium框架。

图像

图3-1 爬虫架构图

爬虫是一门计算机中的技术，它被广泛应用于搜索引擎。常见的搜索引擎有百度、Google、Bing等，其工作原理大致分为爬取信息、存储、建立索引、排序、检索等环节，其中第一阶段就是使用专用程序收集网页数据，这个程序通常称为蜘蛛（Spider）或爬虫（Crawler）。搜索引擎从已知的数据库出发，访问这些网页并抓取文件。搜索引擎通过这些爬虫从一个网站爬到另一个网站，跟踪网页中的链接，访问更多的网页，这个过程称为爬行，这些新的网址会被存入数据库等待搜索。简而言之，爬虫就是通过不间断地访问互联网，然后从中获取用户指定的信息并返回。互联网上，随时都有无数的爬虫在爬取数据，并返回给使用者。

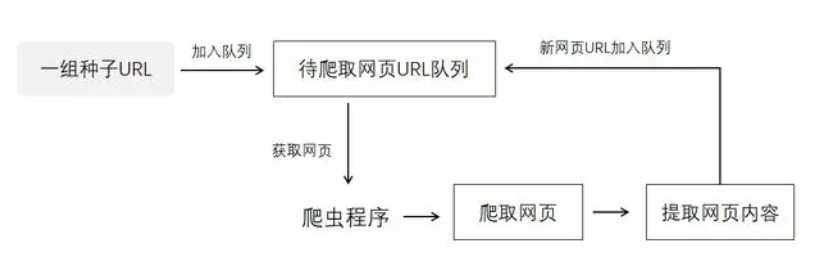


图 3-2 爬虫工作机制

URL就是网页的网址，种子URL就是爬虫要首先爬取的网页网址，确定你的爬虫程序首先从哪些网页开始爬取。一组种子URL是指一个或多个的网页地址。爬虫程序开始工作后，种子URL会先加入到待爬取网页的队列中，爬虫程序从队列按照先进先出的原则获取网页URL，爬虫程序开始爬取网页，爬虫会下载整个网页内容，然后提取网页内容，分析出网页内容包含的URL，并把新的URL加入到队列。当队列为空时，爬虫停止工作，否则爬虫会继续从队列获取网页URL，爬取下一个网页。

爬虫引擎从调度器中取出一个链接(URL)用于接下来的抓取，首先引擎把URL封装成一个请求(Request)传给下载器，然后下载器把资源下载下来，并封装成应答包(Response)，爬虫解析Response解析出实体（Item）,则交给实体管道进行进一步的处理，解析出的是链接（URL）,则把URL交给调度器等待抓取

如果遇到反爬措施比较严格的网站，本文使用selenium进行模拟浏览器，利用浏览器的原生API，封装成一套更加面向对象的Selenium WebDriver API，直接操作浏览器页面里的元素，甚至操作浏览器本身（截屏、窗口大小、启动关闭、安装插件、配置证书等）。Scrapy是一个为了爬取网站数据，提取结构性数据而编写的应用框架。使用Scrapy可以快速搭建爬虫应用程序。掌握Scrapy框架的读者，可以在几天之内就能开发一个爬虫程序，爬取自己需要的数据。

Web Driver本质是一个web服务器，对外提供web api，其中封装了浏览器的各种功能，不同浏览器使用来自不同的Web Driver（由浏览器厂商提供），selenium还可以处理一部分带验证码和点击的情况。

3.2 开源信息数据库

本文所采用的数据集主体是twitter/quora数据集，其中包含了关于美国各大航空公司问题的情绪分析工作。Twitter 数据是从 2015 年 2开始收集的，并要求贡献者首先对正面、负面和中立的推文进行分类，然后对负面原因进行分类（例如“迟到的航班”或“粗鲁的服务”）。Python脚本预处理步骤对收集的数据集执行必要的数据预处理和清理。 在先前收集的数据集上，有一些关键属性文本：tweet本身的文本，created\_at：创建日期，favorite\_count，retweet\_count：收藏和转推的数量，已收藏，已转推：布尔值，指明是否通过身份验证的用户对此推文等有帮助或转发。我们已应用了广泛的预处理步骤，以减小功能集的大小，使其适合于学习算法。本文使用的清洁方法基于字典方法。

从quora/Twitter获得的数据通常包含许多HTML实体，例如＆lt; ＆gt; ＆amp; 嵌入到原始数据中。 因此有必要摆脱这些实体。一种方法是通过使用特定的正则表达式直接删除它们。我们正在使用Python HTML解析器模块，该模块可以将这些实体转换为标准HTML标记。例如＆lt; 转换为“ <”和＆amp; 转换为“＆”。 此后，我们将删除此特殊HTML字符和链接。在解码数据时，这是将信息从复杂的符号转换为简单易懂的字符的过程。收集的数据使用不同的解码形式，例如“拉丁”，“ UTF8”等。本文在进行ML训练前还进行了数据探索。分析总体数据集（twitter，quora和自有标记数据来源之间的关联度分析）。

本文首先使用networkX抓取推特的数据，得到推特2的关键数据。结果表明，网络密度为0.0904，说明网络中的参与者具有一定的相关性。网络中总共有所有节点。共有3287条线，节点之间的最长距离为6，即任意两个节点之间的最长距离为6条线。距离反映了演员之间间隔的长短，距离越小，说明彼此的联系越紧密。也就是说，在推特上，媒体机构之间最多可以通过5个节点相互连接，这比世界上任何两个人之间的平均距离，即距离“六度分离”只有5个节点的距离还要小。理论强调。此外，网络节点之间的平均距离为2.937，这意味着任何两个媒体机构只需要通过1到2个节点的中介即可成为网络中的互联节点。我们的数据集没有特别的富集，比较符合六度原理：（有大v，也有普通网民，之间通过社交网络和进行啦关联，不存在大规模数据评论情绪被操控的情况）。不存在数据偏见，我们可以进行上述清洗、合并之后进行训练。

3.3 文本分析流程

首先是对文本进行清洗，主要是对获取到的数据进行筛选和噪声去除。因为本文使用了爬虫去爬取大量网上的数据，难免存在很多噪声，所以需要一个数据进行筛选和噪声去除，去除掉很多无效信息甚至是干扰信息，因此需要文本处理的过程。

第二步对文本进行分词预处理，在去噪之后，为了更好地进行文本表示，在预处理过程中重要的就是对长文本分局、分词等处理，将大段文本进行分割，分割成以字词为单位的序列。

第三步文本表示，就是提取文本特征将文本表示成向量或矩阵的过程。文本经过清洗和预处理工作后，得到去除了停用词之后的文本分词列表，需要将文本转化数值向量后，进行进一步的处理。

最后是建模分析，采用算法和深度学习模型对文本特征数据进行处理分析。

3.3.1 文本获取与筛选

评论数据具有数据量大、更新快等特点，通过人工查找收集的方式不现实，于是本文在数据获取阶段，选取财经直播网站、评论网站作为文本数据来源，通过网络爬虫收集数据。这些网站有些是传统静态网站，页面内容大部分固定，将一些文本信息封装在网页表格块中，通过正则匹配或XPATH解析的方式，直接从html网页中提取目标文字内容，根据网址的组成规律，实现构建翻页请求、访问新闻或评论详情页链接等一系列操作，从而获取到本文所需要的系列财经文本信息；有些是直播网站，直播网站的页面具有实时更新、滚动加载新内容等特点，与静态网页的信息承载方式有很大不同，因此本文通过对直播界面进行网页源代码分析，发现网页每次刷新后显示的新内容被封装在网页的响应文件中，因此可以通过构造请求，获取并解析响应包。

3.4 数据标注及预处理

文本语料在输送给模型前一般需要一系列的预处理工作, 才能符合模型输入的要求, 如: 将文本转化成模型需要的张量, 规范张量的尺寸等, 而且科学的文本预处理环节还将有效指导模型超参数的选择, 提升模型的评估指标。

本文通过将爬虫爬取到的中文评论通过机器翻译技术翻译成中文，保持原来的样本标签，然后混合几个开源数据集（主要以英文为主），组成本文所用数据集，主要包含环球网各种频道、本地新浪国内新闻、各种资讯等。翻译库使用的是argostranslate库，该库初始化的翻译对象获取从源语言到目标语言的句子列表，语言是两个字母的ISO639代码。该库支持多种语言列表，可以在Hugging Face的模型中心查看所有模型，具体可以翻译的文本长度、运行速度以及它如何响应不寻常的标点符号。数据集主体还是使用啦twitter/quora数据集有：关于美国各大航空公司问题的情绪分析工作。 Twitter 数据是从 2015 年 2开始收集的，并要求贡献者首先对正面、负面和中立的推文进行分类，然后对负面原因进行分类（例如“迟到的航班”或“粗鲁的服务”）。python脚本预处理步骤对收集的数据集执行必要的数据预处理和清理。 在先前收集的数据集上，有一些关键属性文本：tweet本身的文本，created\_at：创建日期，favorite\_count，retweet\_count：收藏和转推的数量，已收藏，已转推：布尔值，指明是否通过身份验证的用户(您)对此推文等有帮助或转发。我们已应用了广泛的预处理步骤，以减小功能集的大小，使其适合于学习算法。 清洁方法基于字典方法。从quora/Twitter获得的数据通常包含许多HTML实体，例如＆lt; ＆gt; ＆amp; 嵌入到原始数据中。 因此有必要摆脱这些实体。 一种方法是通过使用特定的正则表达式直接删除它们。 野兔，我们正在使用PythonHTML解析器模块，该模块可以将这些实体转换为标准HTML标记。 例如＆lt; 转换为“ <”和＆amp; 转换为“＆”。 此后，我们将删除此特殊HTML字符和链接。 在解码数据时，这是将信息从复杂的符号转换为简单易懂的字符的过程。 收集的数据使用不同的解码形式，例如“拉丁”，“ UTF8”等。

关于数据预处理，主要包括停用词的去除和@符的去除。停用词本文使用官方的英文停用词词表，比如”the a”这种，然后遍历整个数据集中的每个单词，如果该单词出现在了停用词列表中，则将其移除，取剩下的部分作为训练和测试集。@符的去除主要通过正则表达式进行搜索，对符合相应模式的句子进行删除，以去除噪声。

分词就是将连续的字序列按照一定的规范重新组合成词序列的过程。我们知道，在英文的行文中，单词之间是以空格作为自然分界符的，而中文只是字、句和段能通过明显的分界符来简单划界，唯独词没有一个形式上的分界符, 分词过程就是找到这样分界符的过程。

下一步数据预处理操作就是将单词转化为数字，这里首先使用keras.preprocessing.text 中的Tokenizer对句子进行分词操作，过滤器包括“!"#$%&()\*+,-./:;<=>?@[\\]^\_`{|}~\t\n”等符号，出现最多的前5个词汇包括及相应频率为“[('flight', 3166), ('not', 1281), ('no', 1226), ('get', 1068), ('t', 981)]”。然后本文将句子转化成相应的独热编码，也就是one-hot向量，用于将文本离散化表示，即初始化一个句子长度的全0list，然后将词表id对应的那个索引置为1，以方便后续神经网络进行建模处理。然后我们将标签也转化为独热编码，用keras中的to\_categorical包，这样方便后续利用交叉熵损失函数对模型进行分类训练。比如将标签0转化为“1，0，0”。

3.5 数据统计及可视化

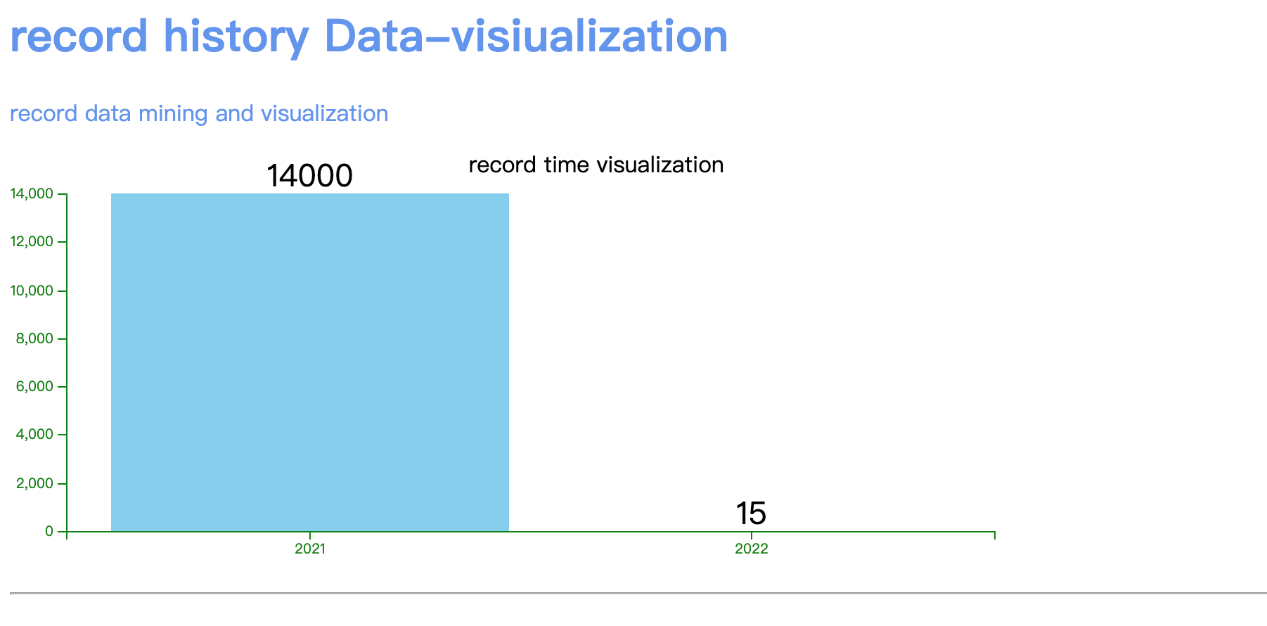


图3-3 数据挖掘和可视化方法记录评论的时间

图3-3展示了爬取的评论数据诞生的时间，可以看到有14000个样本是来自2021年的评论数据，有15个样本是来自2022年的评论数据。图3-4展示了不同标签数据所占的比例，这里使用饼状图进行表示。其中标签为中立的居多，大概占70%，积极的评论其次，大概占20%，所占比例最少的为消极类型的评论，只占总数的10%。关键词相关数据可视化图如图3-5所示，可以看到China关键词所占的评论数据最多，其次是South China Sea，最后是发展关键词。

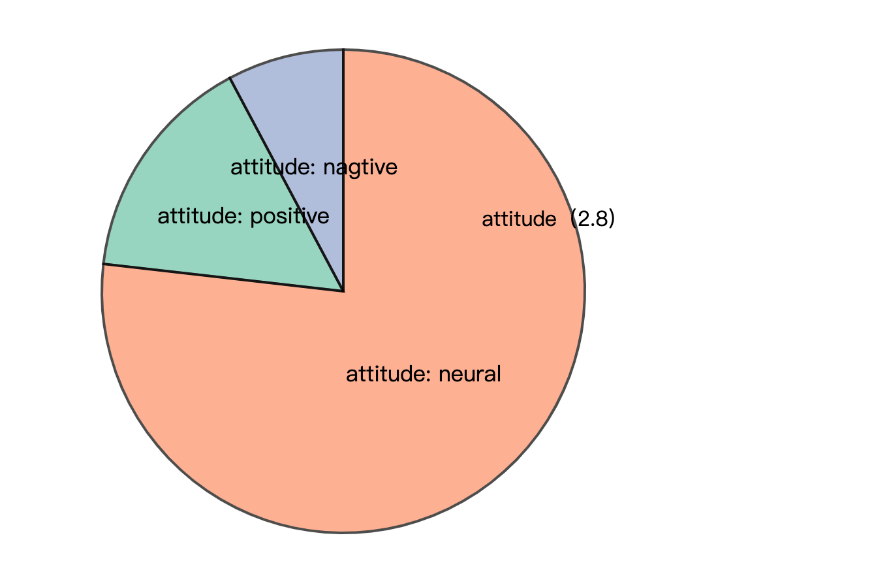


图 3-4 不同标签数据所占比例



图 3-5 关键词相关数据可视化

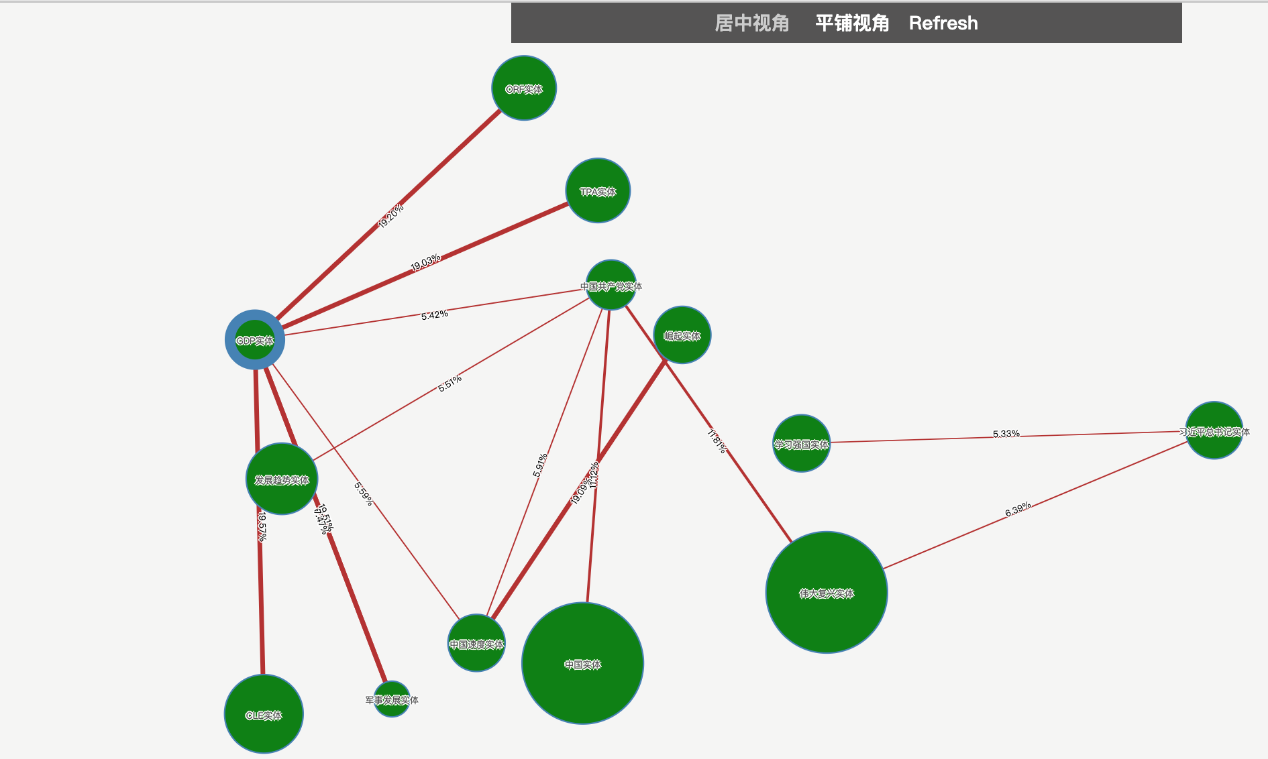


图3-6 知识图谱可视化

图3-6展示了知识图谱的可视化，主要包含了不同实体之间的关联关系。知识图谱主要目标是用来描述真实世界中存在的各种实体和概念，以及他们之间的关系，因此可以认为是一种语义网络。从发展的过程来看，知识图谱是在NLP的基础上发展而来的。知识图谱和自然语言处理NLP有着紧密的联系，都属于比较顶级的AI技术。知识图谱可以用来更高的查询复杂的关联信息，从语义层面理解用户意图，改进搜索质量。知识图谱可用于搜索的终极目标是对万物直接进行搜索。传统搜索是靠网页之间的超链接实现网页的搜索，而语义搜索是直接对事物进行搜索，比如人、物、机构、地点等，这些事物可以来自文本、图片、视频、音频、物联网设备等。知识图谱和语义技术提供了关于这些事物的分类、属性和关系的描述，这样搜索引擎就可以直接对事物进行搜索。比如我们想知道“《觉醒年代》的导演是谁？”，那么在进行搜素时，搜索引擎会把这句话进行分解，获得“《觉醒年代》”，“导演”，再与现有的知识库中的词条进行匹配，最后展现在用面前。传统的搜索模式下，我们进行这样的搜索后得到的通常是包含其中关键词的网页链接，我们还需要在多个网页中进行筛选。可以看出基于知识图谱的搜索更加便捷与准确。也可用于问答，人与机器通过自然语言进行问答与对话也是人工智能实现的标志之一，知识图谱也广泛应用于人机问答交互中。借助自然语言处理和知识图谱技术，比如基于语义解析、基于图匹配、基于模式学习、基于表示学习和深度学习的知识图谱模型。还可用于辅助大数据分析，知识图谱也可以用于辅助进行数据分析与决策。不同来源的知识通过知识融合进行集成，通过知识图谱和语义技术增强数据之间的关联，用户可以更直观地对数据进行分析。此外知识图谱也被广泛用于作为先验知识从文本中抽取实体和关系，也被用来辅助实现文本中的实体消歧，指代消解等。

知识图谱的构建流程一般为首先是获取数据，它们可以是一些表格、文本、数据库等。根据数据的类型可以分为结构化数据、非结构化数据和半结构化数据。结构化的数据为表格、数据库等按照一定格式表示的数据，通常可以直接用来构建知识图谱。非结构化的数据为文本、音频、视频、图片等，需要对它们进行信息抽取才能进一步建立知识图谱。半结构化数据是介于结构化和非结构化之间的一种数据，也需要进行信息抽取才能建立知识图谱。拿到了不同来源的数据时，需要对数据进行知识融合，也就是把代表相同概念的实体合并，将多个来源的数据集合并成一个数据集。这样就得到了最终的数据，在此基础上就可以建立相应的知识图谱了。知识图谱通过知识推理等技术能够获得新的知识，所以通过知识推理可以不断完善现有的知识图谱。

第二步是知识抽取，知识抽取主要针对非结构数据，方法主要包括：实体识别、关系抽取、属性抽取等。首先是针对结构化数据：目前结构化的数据时最主要的知识来源。针对结构化的数据，知识图谱通常可以直接利用和转化，形成基础数据集，再利用知识图谱补全技术进一步扩展知识图谱。其次是针对非结构化数据：针对文本型数据这种非结构化数据，知识获取的方式主要包括实体识别、关系抽取、属性抽取等。具体的方法又包括基于特征模版的方法、基于核函数的监督学习方法、基于深度学习的方法等。关于实体识别，实体识别指在一段文本中识别哪些词代表实体，并打上标签（进行分类）。例如“演员于和伟出演了电视剧《觉醒年代》”这句话中，“于和伟”和“《觉醒年代》”就是两个实体，将他们识别出来之后会分别给“于和伟”打上“演员”的标签，给“《觉醒年代》”打上“电视剧的标签”。然后是关系抽取，识别文本（或其他数据）中实体之间的关系。例如“演员于和伟出演了电视剧《觉醒年代》”这句话中，“出演”为“演员于和伟”与“电视剧《觉醒年代》”之间的关系。

然后是知识融合。当我们想建立一个知识图谱，需要从多个来源获取数据，这些来源不同的数据可能会存在交叉、重叠，同一个概念、实体可能会反复出现，知识融合的目的就是把表示相同概念的实体进行合并，把来源不同的知识融合为一个知识库。知识融合的主要任务包括实体消歧和指代消解，它们都用来判断知识库中的同名实体是代表同一含义、是否有其他实体也表示相同含义。实体消歧专门用于解决同名实体产生歧义的问题，通常采用聚类法、空间向量模型、语义模型等。指代消解则为了避免代词指代不清的情况。

最后是知识推理，基于知识图的知识推理旨在识别错误并从现有数据中推断新结论。通过知识推理可以导出实体间的新关系，并反馈以丰富知识图，从而支持高级应用。鉴于知识图的广泛应用前景，大规模知识图的知识推理研究成为近年来自然语言处理领域的一个研究热点。可以看出，在NLP领域，知识图谱的应用主要集中在搜索和推荐领域，问答系统（其本质也是搜索和推荐的延伸）。在语义搜索这一块，知识图谱的搜索不同于常规的搜索，常规的搜索是根据keyword找到对应的网页集合，然后通过Page Rank等算法去给网页集合内的网页进行排名，然后展示给用户；基于知识图谱的搜索是在已有的图谱知识库中遍历知识，然后将查询到的知识返回给用户，通常如果路径正确，查询出来的知识只有1个或几个，并且相当精准。在问答系统中，系统同样会首先在知识图谱的帮助下对用户使用自然语言提出的问题进行语义分析和句法分析，进而将其转化成结构化形式的查询语句，然后在知识图谱中查询答案。

3.6 本章小结

第4章 基于深度学习的文本分析

4.1 多模型情感分类器训练

本文所使用的深度学习模型为全连接神经网络。人工神经网络（Artificial Neural Network, ANN）简称神经网络，可以对一组输入信号和一组输出信号之间的关系进行模拟，是机器学习和认知科学领域中一种模仿生物神经网络的结构和功能的数学模型。用于对函数进行估计或近似，其灵感来源于动物的中枢神经系统，特别是大脑。神经网络由大量的人工神经元（或节点）联结进行计算，大多数情况下人工神经网络能在外界信息的基础上改变内部结构，是一种自适应系统。具有 n 个输入一个输出的单一的神经元模型的结构中，神经元接受来自 n 个其他神经元传递过来的输入信号，这些输入信号通过带权重的连接进行传递，神经元收到的总输入值将经过激活函数 f 处理后产生神经元的输出。全连接神经网络（Multi-Layer Perception, MLP）或者叫多层感知机，是一种连接方式较为简单的人工神经网络结构，属于前馈神经网络的一种，只要有输入层、隐藏层和输出层构成，并且在每个隐藏层中可以有多个神经元。MLP 网络是可以应用于几乎所有任务的多功能学习方法，包括分类、回归，甚至是无监督学习。神经网络的学习能力主要来源于网络结构，而且根据层的数量不同、每层神经元数量的多少，以及信息在层之间的传播方式，可以组合成多种神经网络模型。全连接神经网络主要由输入层、隐藏层和输出层构成。输入层仅接收外界的输入，不进行任何函数处理，所以输入层的神经元个数往往和输入的特征数量相同，隐藏层和输出层神经元对信号进行加工处理，最终结果由输出层神经元输出。根据隐藏层的数量可以分为单隐藏层 MLP 和多隐藏层 MLP。针对单隐藏层 MLP 和多隐藏层 MLP，每个隐藏层的神经元数量是可以变化的，通常没有一个很好的标准用于确定每层神经元的数量和隐藏层的个数。根据经验，更多的神经元就会有更强的表示能力，同时更容易造成网络的过拟合，所以在使用全连接神经网络时，对模型泛化能力的测试很重要，最好的方式实在训练模型时，使用验证集来验证模型的泛化能力，且尽可能地去尝试多种网络结构，以寻找更好的模型，但这往往需要耗费大量的时间。与传统的感知机不同，每个结点和下一层所有结点都有运算关系，这就是名称中“全连接”的含义，上图的中间层也成为隐藏层，全连接神经网络通常有多个隐藏层，增加隐藏层可以更好分离数据的特征，但过多的隐藏层也会增加训练时间以及产生过拟合。

全连接神经网络还使用反向传播技术来实现参数的更新。我们可以将反向传播算法定义为在已知分类的情况下，为给定的输入模式训练某些给定的前馈神经网络的算法。当示例集的每一段都显示给网络时，网络将查看其对示例输入模式的输出反应。之后，测量输出响应与期望输出与误差值的比较。之后，我们根据测量的误差值调整连接权重。

反向传播算法最早出现在20世纪60年代，30年后由大卫·鲁梅尔哈特、杰弗里·辛顿和罗纳德·威廉姆斯在1986年的著名论文中推广。在这篇论文中，他们谈到了各种神经网络。今天，反向传播做得很好。神经网络训练是通过反向传播实现的。通过这种方法，我们根据前一次运行获得的错误率对神经网络的权值进行微调。正确地采用这种方法可以降低错误率，提高模型的可靠性。利用反向传播训练链式法则的神经网络。简单地说，每次前馈通过网络后，该算法根据权值和偏差进行后向传递，调整模型的参数。典型的监督学习算法试图找到一个将输入数据映射到正确输出的函数。反向传播与多层神经网络一起工作，学习输入到输出映射的内部表示。损失函数为一个或多个变量被映射到实数，这些实数表示与这些变量值相关的某个数值。为了进行反向传播，损失函数计算网络输出与其可能输出之间的差值。

本文所使用的模型的网络架构如图4-1所示，其由4个全连接层和1个dropout层组成。全连接层是为了对输入的特征向量进行降维，dropout层是为了防止过拟合，对部分神经元随机地丢掉，以使得模型的泛化能力更强，也就是说不是仅仅去迎合训练数据的特征，而是在测试数据上也能表现出良好的泛化效果。输入层的神经元个数为10000，因为本文设置的词表大小为10000，因此在将文本进行分词处理后，每个词都转化为一个在词表中的id。词表的建立是通过训练集中每个单词出现的词频进行建立的，设置为10000是遵循了经典的自然语言处理设置模式。将每个词在词表中找到相应的id后，对id进行独热编码（one-hot），使得其可以表示离散的文本特征，然后再通过word2vec即可将离散的编码转化为连续的编码。然后输入进全连接神经网络即可进行特征提取以进一步做最终的分类。最终的分类层设置的神经元个数为3，是因为最终的标签有三类：中立、积极和消极。模型最终的输出为输入的句子向量属于这三类标签的概率。模型训练使用的损失函数为交叉熵损失函数，是用于分类问题中最为常见的损失函数。如图4-2所示，整个模型的参数量大小为320131，这些全部都是可训练的参数。

关于基线模型，本文使用全连接神经网络，第一个基线模型一共有3层，第一层的隐藏神经元个数为64，第二层的隐藏神经元个数为64，第三层的神经元个数为3，即本分类问题对应的标签个数。训练过程使用的优化器是“rmsprop”，损失函数使用的是“categorical\_crossentropy”，模型的定义通过keras中的model.compile来实现。模型的训练部分使用model.fit来实现，指定好训练数据和验证数据。模型训练的周期为20，训练的batch\_size大小为512。评估指标为accuracy，即分类问题的精确度。

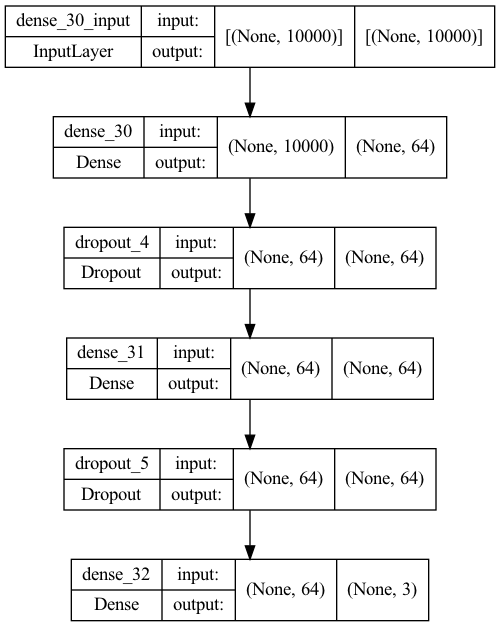


图4-1 训练情感分类的神经网络架构图

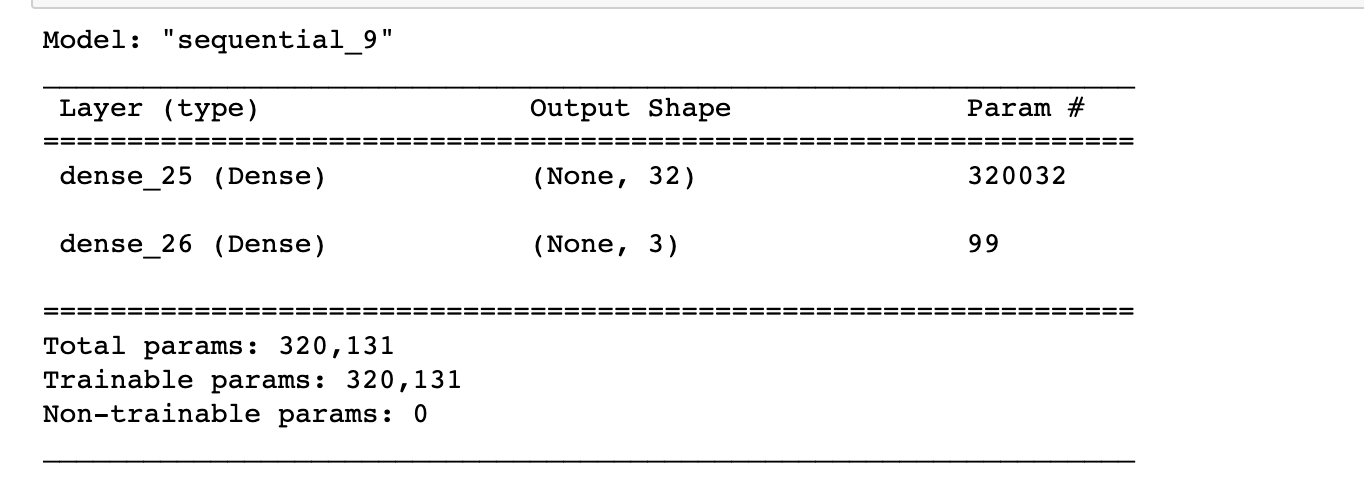


图4-2 基线模型一网络架构及参数量大小展示

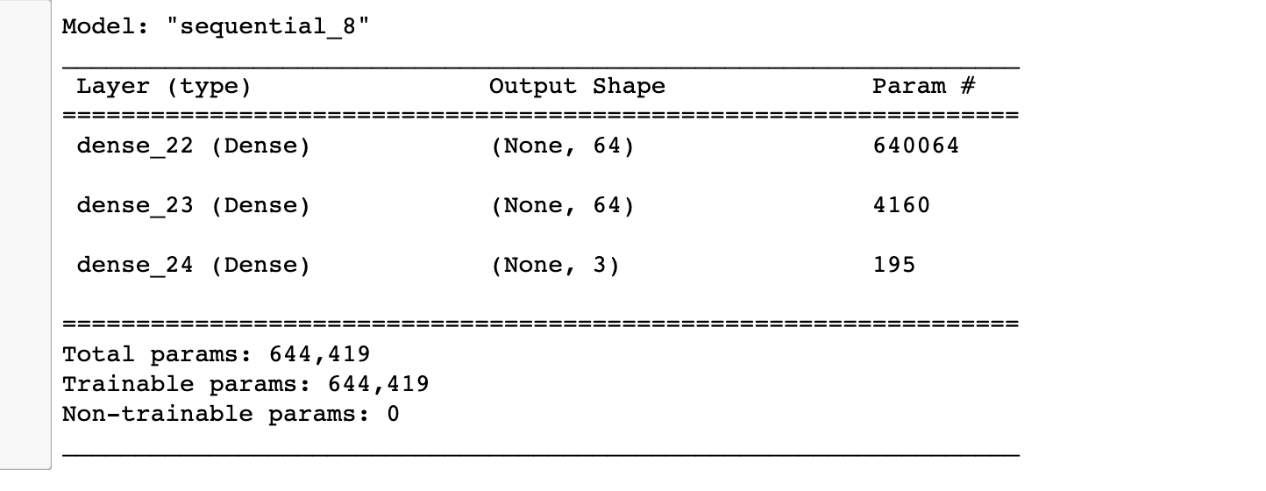


图4-3 基线模型二网络架构及参数量大小展示

4.2 数据集与分类模型性能分析

数据集一共包含14640个句子，本文按照训练和测试集8：2的比例进行划分，具体使用的是sklearn.model\_selection包中的train\_test\_split包。数据集通过pandas对csv数据进行读取，数据的样例如图4-4所示。最左边一列代表标号，text一列代表英文句子，airline\_sentiment一列代表相应的情感标签，分为negative、neural和positive三大类。

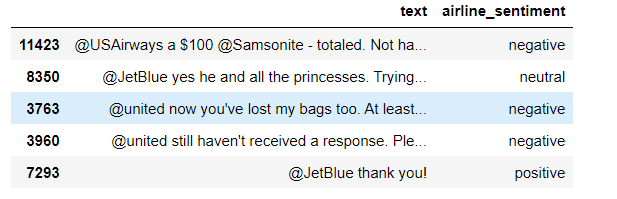


图4-4 数据集样本展示

基线模型使用的是三层全连接网络，其参数量较少，在拟合上万条数据的时候容易存在过拟合的问题，因此本文对齐进行了两种优化方式：第一种是采用L2正则化的方法，第二种是采用dropout机制。二者对神经网络的优化均为防止神经网络过拟合的目标而实现。L2正则化方法的物理含义就是在代价函数后面加上一个正则化项以约束模型参数的空间范围，也是对模型的参数设定一个先验（来自于贝叶斯学派的观点）。Dropout机制本质上就是对每一层的神经元做随机丢弃，以防止模型过拟合训练数据，而在测试数据或者是验证数据上表现的效果较差。通过引入L2正则化和Dropout机制对模型全连接神经网络进行优化，可以有效地提升模型执行情感分类任务的效果。

4.3 实验结果对比与分析

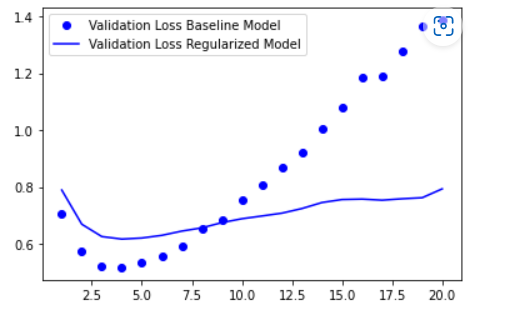


图4-5 全连接网络引入L2正则化后的loss对比图

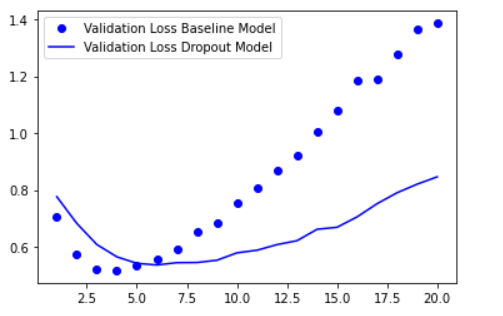


图4-6 全连接网络引入dropout机制后的loss对比图

图4-5为引入了L2正则化前后模型的loss对比图，可以看到引入了L2正则化后，模型在验证集上的loss明显低于引入之前，而loss函数反映了模型在数据上与真实标签的差异，越小越好，因此说明了L2正则化方法对于提升模型性能的有效性。图4-6为引入了Dropout机制后，模型在验证集上的loss与引入之前的对比，可以看到，模型的loss要明显低于引入之前的loss，因此可以得出结论，引入L2正则化机制和Dropout机制均可明显提升模型的分类效果。

表X汇报了引入Dropout机制的模型与基线模型在测试集上的准确率对比结果。

4.4 本章小结

第5章 系统界面设计

5.1 UI界面设计及美化

本文做了一个后端api服务，在一个后端webserver将训练好的模型部署起来（wsgiserver/flask api等）。本文的前后端主要基于：flask+sqlchemy + numpy+ html5+vue+jquery技术栈。首先，把第一步训练好的机器学习模型和参数缓存起来，然后放在flask同级别地方方便调用。然后，本文把第一步的机器学习模型训练好的模型缓存起来，拷贝到 api后端工程，集成在一起，提供接受输入，使得模型进行分类预测的api。对于科学数据集的管理，本文使用flask 后端和flask cros ，前端使用vue、bootstrap和 jinja，在前端接收了数据后，传递到后端sqlchemy进行存储。整体上主要是前端Jinja的template页面触发事件，然后通过Flask API传递到android 预测client进行调用。

Flask 是一个基于 Python 的轻量级 Web 框架，WSGI 工具箱采用 Werkzeug，模板引擎使用 Jinja2。由于其不依赖于特殊的工具或库，并且没有数据抽象层、表单验证或是其他任何已有多种库可以胜任的功能，从而保持核心简单、易于扩展，而被定义为"微"框架。但是，Flask 可以通过扩展来添加应用功能。并且 Flask 具有自带开发用服务器和 debugger、集成单元测试和 RESTful 请求调度 (request dispatching)、支持 secure cookie 的特点。本文主要使用Flask的网站部分和wsgi写API部分。Flask主要包括Werkzeug和Jinja2两个核心函数库，他们分别负责阢处理和安全方面的工鞥呢，这些基础函数为Web项目开发过程提供了丰富的基础组件。Flask中的Jinja2模板引擎，提高了前端代码的复用率。可以大大提高开发效率并且有利于后期的开发与维护。Flask不会指定数据库和模板引擎等对象，用户可以根据需要自己选择各种数据库。Flask不提供表单验证功能，在项目实施过程中可以自由配置，从而为应用程序开发提供数据库抽象层基础组件，支持进行表单数据合法性验证、文件上传处理、用户身份认证和数据库集成等功能。总体而言，Flask的特点可以概括为：因为灵活，轻便高效，被业界所认可，同时拥有基于Werkzeug、Jinja2等一些开源库，拥有内置服务器和单元测试，适配RESTful。我们使用flask编写网站的用户登录/注册/权限管理/个人主页/机器学习训练和可视化的前后台逻辑部分，非常方便后续进行扩展。本文将使用 SQLite，这是一个小型 SQL 数据库实现，非常容易启动和运行。此外，本文在 Flask 项目中设置 SQLAlchemy，本文导入 flask\_sqlalchemy 软件包，然后将 Flask app 变量包装在新的 SQLAlchemy 对象。本文还在 Flask 应用程序配置中设置 SQLALCHEMY\_DATABASE\_URI 以指定本文要使用的数据库以及如何访问它。

Jinja2 是基于 Python 的模版引擎，支持 Unicode，具有集成的沙箱执行环境并支持选择自动转义。Jinja2 拥有强大的自动 HTML 转移系统，可以有效的阻止跨站脚本攻击；通过模版继承机制，对所有模版使用相似布局；通过在第一次加载时将源码转化为 Python 字节码从而加快模版执行时间。本文的网站看得到的页面部分是这块开发。

关于API风格，本文可以开始定义 RESTful 处理程序。我们将使用 Flask-RESTful 软件包，这是一组工具，可帮助我们使用面向对象的设计来构建 RESTful 路由。REST 系统的特点主要包括以下几点：

1.客户端-服务器: 客户端和服务器之间隔离，服务器提供服务，客户端进行消费。

2.无状态: 从客户端到服务器的每个请求都必须包含理解请求所必需的信息。换句话说， 服务器不会存储客户端上一次请求的信息用来给下一次使用。

3.可缓存: 服务器必须明示客户端请求能否缓存。

4.分层系统: 客户端和服务器之间的通信应该以一种标准的方式，就是中间层代替服务器做出响应的时候，客户端不需要做任何变动。

5.统一的接口: 服务器和客户端的通信方法必须是统一的。

6.按需编码: 服务器可以提供可执行代码或脚本，为客户端在它们的环境中执行。

此外，本文需要设置 Flask-RESTful 扩展名才能在 Flask 服务器中启动并运行。Flask-RESTful 是一个 Flask 扩展，它添加了快速构建 REST APIs 的支持。它当然也是一个能够跟你现有的ORM/库协同工作的轻量级的扩展。Flask-RESTful 鼓励以最小设置的最佳实践。

关于前端，整个平台的前端部分和可视化部分本文主要是使用vue、jquery和html5。Vue 是一套用于构建用户界面的渐进式 JavaScript 框架 ；同时它是一个典型的 MVVM 模型的框架（即：视图层-视图模型层-模型层）;HTML5是HTML的新标准，是一种超文本标记语言，是用来创建网页的标准标记语言，通过一系列的标识，来规范网络上的文档格式。Vue是一个渐进式 JavaScript 框架，而HTML5是一种超文本标记语言。其次，在开发中vue框架通过mvvm的模式，解耦了视图层与模型层，而HTML5原生开中数据与标签紧耦合；但是vue和html5可以进行结合。vue是一个前端框架，但还是建立在HTML，CSS，JavaScript的基础之上的，通过编译之后依然是HTML、CSS和JavaScript组成。

本文所设计的平台界面如图5-1所示，其名称为“情绪检测大数据平台”，主要支持的功能有用户管理、历史数据可视化（柱状图形式和饼状图形式）、知识图谱可视化、注册和登录，和机器学习接口（ML）等。主界面还展示了测试接口，以及两个新闻标题和正文。同时，平台支持搜索功能，用户可以根据关键词搜索定位到用户想使用的模块，如果不包含该模块，则输出页面不存在。



图5-1 平台主界面展示

用户管理界面如图5-2所示，其展示了用户名，缩写名，角色和相应的操作。该页面为管理员管理用户数据库列表所用，支持的操作包括查看、修改和删除。查看即为查看该用户的具体资料信息，登录访问的历史轨迹等；修改为修改该用户的使用权限；删除为在数据库列表中删除该用户。



图5-2 用户管理界面

注册界面如图5-3所示，主要包括待输入的注册用户名或邮箱，要设置的密码和密码确认三个弹窗部分。本文通过字符串匹配机制判断用户输入的待设置密码和确认密码是否一致，如果不一致则弹出窗口要求用户重新输入密码，否则提示注册成功，将用户信息加入数据库，并及时在后台管理系统更新。

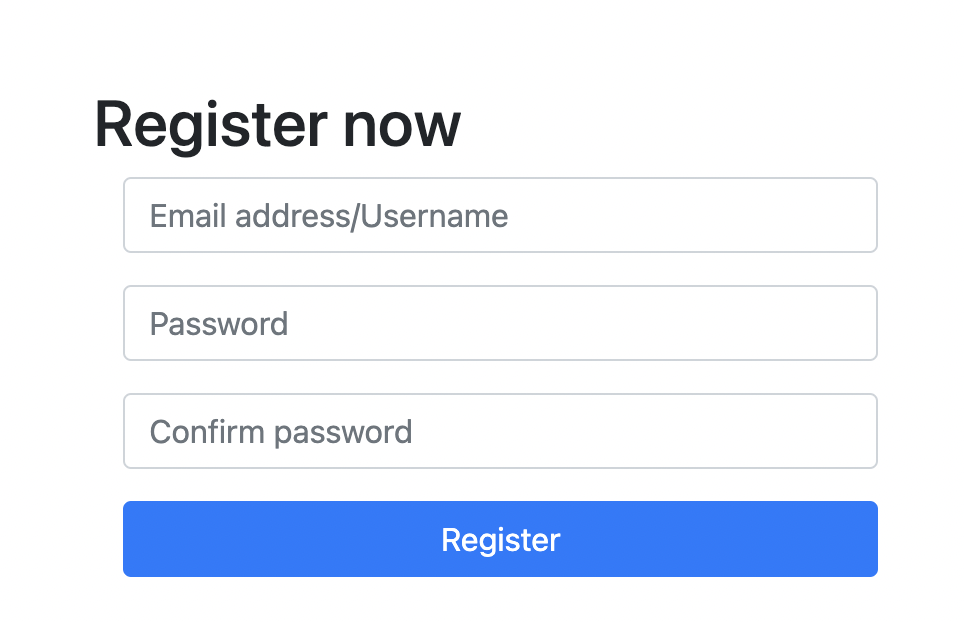


图5-3 用户注册界面

用户登录界面如图5-4所示，在UI界面的右上角，分别提示用户输入相应的邮箱地址和密码，然后点击绿色按钮sign in即可登录，如果用户名和密码与数据库中存储的key-value键值对匹配，则提示用户登录成功，否则提示登录失败。如果用户名不在数据库列表中，则提示无效用户名，重新输入。



图5-4 用户登录界面

5.2 手机App设计与系统互联

本文还设计了安卓app情感监测版本，通过调用库分析某些本地库权限list，然后app端根据获取的权限，动态传回机器学习api接口，然后获得预测结果，进行不同的弹框提示。安卓界面的主界面如图5-5所示，主要包括标题“Detect Sentiment by ML”，和“查看监测结果”的按钮，通过上传要做情感分类分析的句子后，即可通过点击查看监测结果按钮得到答案。

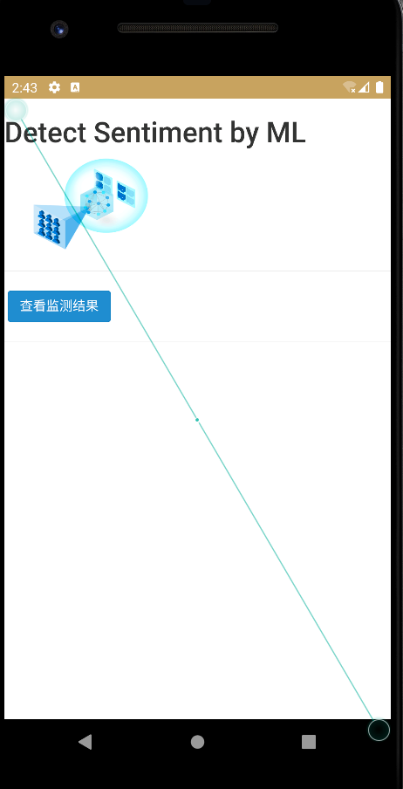


图 5-5 安卓端界面展示

具体的情感监测案例如图5-6所示，也称作网络信息监测模块，主要由蓝色部分的句子和红色部分的标签组成。其中，1.0表示积极评论，0.0表示消极评论。比如“我很好”一句为积极的评论，则模型输出1.0，表示积极态度。而“美国艾奥瓦州山路地区出现山火”为一句消极的评论，因此模型输出0.0，表示消极的态度。

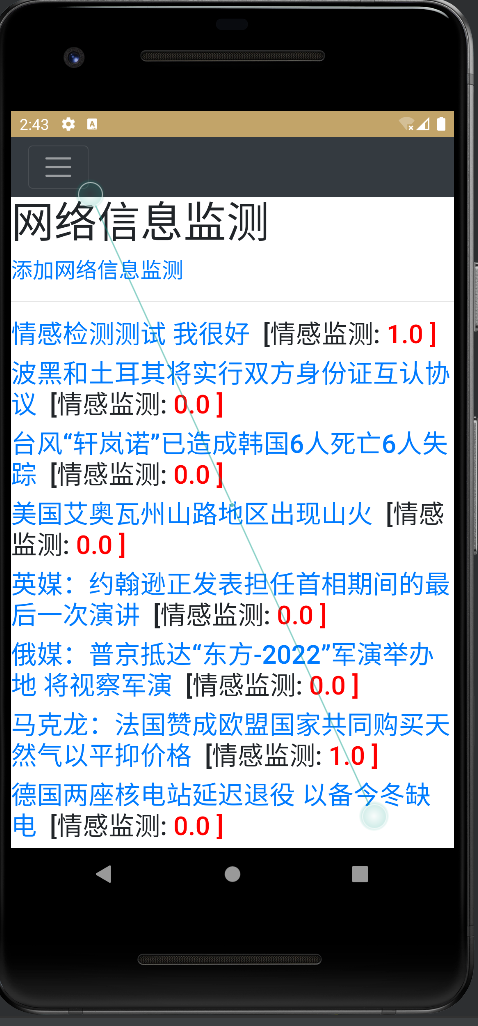


图5-6 安卓端网络信息检测和情感检测结果展示图

第6章 全文总结与展望

6.1 全文总结

6.2 后续工作展望

致谢

不知不觉中，已经快到了和母校说再见的时候。回首过去X年半在母校的时光，我不仅学会了知识，结识了恩师，收获了友谊，还经历了一些挫折，学会了很多做人做事的道理。我感谢母校，感谢这段时光所经历的磨练，这是我人生的一笔宝贵财富，使我更能自信和勇敢地面对未来的工作和生活。

首先，我要对我的导师—XX老师致以真诚的感谢和敬意。感谢X老师在我XX阶段对我的教导和关心，帮助我开辟了一条科研学术的道路，让我知道为何要终身学习，为何要锻炼大脑、锻炼思维模式。X老师对我们的工作要求一致很严格也很关心，就是因为这份严格和细致，促使我们不断向优秀的人优秀的团队看齐，不断地以严要求高标准对待自己的学习和工作。我佩服X老师的睿智、严谨以及对真理对学术不懈的热爱和追求，这些将成为我今后学习和生活的榜样，不断鞭打我变得更加成熟和优秀。

同时，我还要感谢师门的各位兄弟姐妹。在学术上，他们为我指引道路，同他们的交流能够激发我的灵感和新思路。感谢XXX。与他们的相处就像一个大家庭一样，让我感受到了温暖。

参考文献

1. Adam, Klaus, Albert Marcet, and Juan Pablo Nicolini. "Stock market volatility and learning." The Journal of finance 71.1 (2016): 33-82.
2. O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458 (2015).
3. Medsker, Larry R., and L. C. Jain. "Recurrent neural networks." Design and Applications 5 (2001): 64-67.
4. Noble, William S. "What is a support vector machine?." Nature biotechnology 24.12 (2006): 1565-1567.
5. Hecht-Nielsen, Robert. "Theory of the backpropagation neural network." Neural networks for perception. Academic Press, 1992. 65-93.
6. Yu, Yong, et al. "A review of recurrent neural networks: LSTM cells and network architectures." Neural computation 31.7 (2019): 1235-1270.
7. Huang, Zhiheng, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging." arXiv preprint arXiv:1508.01991 (2015).
8. “Yahoo finance,” accessed: 2020-03-12. [Online]. Available: https:

//finance.yahoo.com/quote/GOOG/history?p=GOOG.

1. Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." arXiv preprint arXiv:1803.08375 (2018).