南京大学

本 科 毕 业 论 文

院　　系　计算机科学与技术

专　　业　计算机科学与技术

题　　目　一个热点事件舆情分析系统的构建

年　　级　16　学　　号　161220017

学生姓名　陈翔

指导老师　陈家骏　职　　称　教授

提交日期　2020 年 5 月 29 日

南京大学本科生毕业论文（设计）中文摘要

毕业论文题目：一个热点事件舆情分析系统的构建

计算机科学与技术 院系 计算机科学与技术 专业

16 级本科生姓名：陈翔 指导教师（姓名、职称）：陈家骏 教授

摘要：

当前社会正处于快速发展的新时期，各行各业的热点事件层出不穷，而利用人工智能等领域技术把握网民的舆论动向逐渐成为热门需求。本毕业设计主要围绕如何实现一个舆情分析系统展开，首先利用爬虫搜集互联网上相关事件的博客、文章、评论等信息并进行去噪、去重、分词、筛选、向量化等预处理，其次利用各种机器学习方法，包括朴素贝叶斯分类器、支持向量机、多层感知机、集成学习器等，训练并且验证清洗好的数据，比较它们之间的效率与效果，最后选取其中表现最好的分类器对标签未知的数据进行预测。

关键词：机器学习；网络爬虫；舆情分析；数据挖掘；情感分析

南京大学本科生毕业论文（设计）英文摘要

THESIS: Construction of a public opinion analysis system for hot events

DEPARTMENT: Computer Science and Technology

SPECIALIZATION: Computer Science and Technology

UNDERGRADUATE: Xiang Chen

MENTOR: Prof. Jiajun Chen

ABSTRACT:

At present, society is in a new period of rapid development. Hot events in various industries are emerging in an endless stream, and the use of artificial intelligence and other technologies to grasp the trend of public opinion of netizens has gradually become a hot demand. This graduation project mainly focuses on how to implement a public opinion analysis system. First, crawlers are used to collect information such as blogs, articles, and comments on the Internet, and preprocessing such as denoising, deduplication, word segmentation, screening, and vectorization is performed. A variety of machine learning methods, including naive Bayes classifiers, support vector machines, multi-layer perceptrons, integrated learners, etc., train and verify the cleaned data. We compare the efficiency and effectiveness between them, and finally select the best performing one to predict data with unknown labels.

KEY WORDS: Machine Learning; Network Crawlers; Public Opinion Analysis; Data Mining; Emotion Analysis

目 录

第 1 章　引言

近年来网络舆论发酵愈发迅速，而一个热点事件的舆情分析系统能够即时地从中挖掘出有效的信息，为决策的制定提供依据。相比于传统的人工方法，舆情分析系统的自动化等特点节省了大量的人力物力，逐渐成为了研究热点。

1.1　研究背景

当前社会正处于快速发展的新时期，各种社会热点事件层出不穷，比如最近的新冠状病毒肺炎引起了全球性感染，对我国乃至全世界人民的日常生活造成了巨大的影响，而在信息时代的大背景下，人们最先想到的发声渠道往往是各大网络平台。如何快速掌握群众意见并据此做出相应的分析与决策，逐渐成为了许多人、机构乃至政府机关的迫切需求。

舆情，或称舆论、民意，维基百科对其的定义是“在一定社会范围内，消除个人意见差异，反映社会知觉和集体意识的、多数人的共同意见”。随着互联网与人们联系日渐紧密，越来越多的社会热点可以激发人们的发声欲，而在中文互联网上，这样的发声平台基于用户间的关系可分为两种，一种是弱关系平台，包括微博、知乎、虎扑、豆瓣等，另一种则为强关系平台，包括微信朋友圈、QQ 空间和人人网。对于后者，人们除了对公共事件发表看法，更多地是在分享生活中的细节，此外获得信息的渠道也较为局限，因此本文将目光主要聚焦在前者。舆情分析系统是一种对舆论进行采集、处理并分析的工具。一般来说，一个成熟的舆情分析系统需要拥有快速采集数据、接收海量数据、处理多种语言、精准分析情绪、高处理效率、高度可视化等能力 [1]。

1.2　研究现状

目前国内市面上有不少成熟的舆情分析系统，比如新浪舆情通，百度舆情，企鹅风讯和人民舆情等等，前三者依托互联网大厂拥有天然的流量优势和技术积累，而人民舆情则背靠人民日报，在内容生产方面优势明显。

在舆情分析中，除了必不可少的获取原始信息的过程，大量的工作都集中在数据挖掘上，而数据预处理又占了数据挖掘相当大的比重。Alasadi 等人对数据预处理使用到的技术做了一次回顾，主要包括数据清理、集成、转换和归约等 [2]。数据清理的主要任务包括缺失值处理和去噪，其中去噪包括重复数据的清除。1970 年布隆提出了一个检索某个元素是否在集合中的方法，称之为布隆过滤器，该方法组合了多种性能优良的哈希函数，在空间和时间效率上远超一般的同类算法，广泛应用于实际的去重场景中，其中不乏微软 [3]、谷歌 [4][5] 等科技巨头的身影。数据集成的主要目的是集成多个数据源的数据，提高数据的完整性。数据转换则是将数据转换成后续的算法能够接受的格式，对于中文文本分析来说，这一步主要包括分词和向量化。分词是非拉丁语系语句在进行自然语言处理任务时特有的阶段，主要有两种实现原理——基于字符串匹配和基于统计。前者按照一定的策略将待匹配的字符串和一个已建立好的词典中的词进行匹配；后者则利用统计机器学习模型对大量已分词的文本进行训练，从而实现对未知文本的切分，该方法最早是由 Nianwen Xue 于 2002 年提出 [6] 的。词的向量化则是要将词转换成数字，以方便计算机处理。最简单的方法是直接统计每个词出现的频率，称之为词袋模型，但是这种方法会偏重长文本，因此又引入了 tf-idf 模型，该模型还考虑了文档频率，其相反的概念 idf（逆文档频率）最早由 Hans 于 1957 年提出 [7]。最后一种考虑上下文的方法称为词嵌入模型，其中最著名的是 Word2vec，它是一个双层神经网络，以一个大型文本语料库为输入，训练后产生一个通常拥有几百维度的向量空间，语料库的每个单词都在该空间中对应唯一一个向量。该模型是由 Tomas Mikolov 领导的团队于 2013 年提出并发表的 [8]。数据归约是在不丢失主要信息的情况下降低数据维度，主要方法有采样、PCA（主成分分析）和 SVD（奇异值分解）。Jacob 等人在 2018 年发布了 BERT 预训练模型，它是一个基于微调的深度学习模型，通过使用遮蔽语言模型来实现预训练的深度双向表示，刷新了 11 项 NLP 任务的性能记录 [9]。

舆情分析系统的另一个重要维度是文本情感分析。文本情感分析一直是 NLP 领域的大热门，这一工作首先需要甄别文本来源，过滤掉不带感情色彩的客观性文本，提取主观性文本，其次需要对主观性文本进行分析，主要包括文本情感极性分析和极性强度分析。按照文本的颗粒度，文本情感分析可分为词、句子、篇章三个层次的识别与分析。词汇级情感分析的基础工作是情感词的抽取，分为基于语

料库和基于词典两种研究方法，前者利用大语料库的统计特性，简单易行，后者获取的情感词较为全面，但存在歧义现象。语料库一般通过人工标注构成，但也有人提出自动标注的方法，缺点是精确度不高，情感词典则主要采取人工方法生成。早期曾有基于简单统计的情感倾向分类，比如 Tsou 等利用大众对名人的评价统计分析了极性元素的分布密度和语义强度以得到词语的语义倾向 [10]。后来机器学习方法兴起，Pang 等利用词袋模型、朴素贝叶斯分类器和支持向量机对电影影评进行情感倾向分析 [11]。Taboada 则提出基于词库的方法，对带有一定倾向和强度的情感词的词典采用集约化方法去计算每个文本的情感分值，最终确定文本的情感倾向 [12]。本文根据采用的语料库的特点，将文本分为正负两个极性的情感，同时主要针对句子这一层级进行情感分析。

本文通过借鉴上述工作，试图实现一个较为完整的舆情分析系统，包括从网络获取数据，数据预处理，训练和比较分类器并预测文本感情倾向等。

1.3 论文的主要工作

本文实现的舆情分析系统主要包括三个模块，分别是数据爬取模块、数据预处理模块和情感分析模块。其中，数据爬取模块会从新浪微博中爬取大量原始信息，通过分析、筛选、去噪等一系列手段将其中有效信息以一定格式储存下来。数据预处理模块则会对有效信息进一步清洗，包括去重、分词、删词、向量化等过程，目标是将文本信息转换成为分类器可以接受的实数矩阵形式。情感分析模块将会提供不同的分类器的选择，比如朴素贝叶斯分类器、支持向量机、多层感知机等，本文将会通过定性和定量分析厘清它们之间的优劣势，并选择最合适的分类器对测试文本进行情绪预测。

1.4 论文的组织

论文分为四个章节。第一章为引言，介绍了本文的研究背景、现状和论文结构等；第二章为背景知识介绍，介绍了系统构建过程中用到的的基础知识；第三章为舆情分析系统的构建，介绍了系统实现的细节并展示了实验结果；第四章总结了主要工作，并讨论了需要改进的地方。

第 2 章　背景知识介绍

本章将介绍本文的系统构建中用到的一些基础知识和工具。本文假设读者拥有基本的微积分、线性代数和概率论知识。

2.1　布隆过滤器

布隆过滤器是一种数据结构，由一个很长的二进制向量和一系列随机哈希函数，一般用于检索一个元素是否在一个集合中。它的优点在于空间效率和时间复杂度都远远优于一般算法，缺点是可能存在误判，具体来说，它可能把一个不属于该集合的元素误判为属于该集合，即可能会存在假阳性匹配，但一定不会存在假阴性匹配。在参数合适的情况下，出现假阳性匹配的概率非常小，因此该算法往往效果不错 [13]。

布隆过滤器的原理是，当一个元素被加入集合时，用 k 个哈希函数将这个元素映射成一个二进制数组中的 k 位，并将其置 1，检索时，需要看待检索元素被映射的 k 位是否都是 1 即可，若有至少 1 位为 0 则一定不在集合中，否则很可能在集合中。注意到，由于布隆过滤器只保存一个二进制向量，因此后面添加的元素映射的位很可能覆盖之前的位，这是允许的，也是需要很长的二进制向量的原因；否则，很容易就能将二进制向量的大部分位数置 1，使得误判率大幅上升。

接下来需要推导出合适的参数。假设给定元素数目 n 和能接受的假阳性概率 p，并假设所有哈希函数以相等概率映射每一位，需要计算的是最优数组长度 m 和最优哈希函数个数 k。易知数组中某一位在某一个元素的一次哈希映射中没有被置 1 的概率为 $1 - \frac{1}{m}$，则在 k 次哈希中都没被置 1 的概率为 $(1 - \frac{1}{m})^k$，则在插入 n 个元素后该位仍为 0 的概率为 $(1 - \frac{1}{m})^{kn}$，因此该位为 1 的概率为 $1 - (1 - \frac{1}{m})^{kn}$。对于某一个不在集合中的元素，它被误判认为在集合中的概率则为

$$(1 - (1 - \frac{1}{m})^{kn})^k \doteq (1 - e^{-kn/m})^k.$$

容易看出，随着数组长度 m 的增加，或者元素数目 n 的减小，假阳性的概率都会

下降。利用求极值的方法，发现对于给定的 m 和 n，当

$$k = \frac{m}{n}\ln 2$$

时假阳性概率最低，令此时的假阳性概率为 p，则有

$$p = ((1 - e^{-(\frac{m}{n}\ln 2)n/m})^{\frac{m}{n}\ln 2}),$$

该式可化简为

$$\ln p = -\frac{m}{n}(\ln 2)^2,$$

因此

$$m = -\frac{n\ln p}{(\ln 2)^2}. \tag{2.1}$$

2.2 Tf-idf

Tf-idf 是一种统计方法，用以评估某一词汇对于一个文档的重要程度 [14]，其计算方法如下：

$$\mathrm{tf}(t,d) = \frac{\mathrm{occu}(t,d)}{\sum_{k=1}^{n}\mathrm{occu}(t,k)}$$

$$\mathrm{idf}(t) = \log\frac{1+n}{1+\mathrm{df}(t)} + 1$$

$$\text{tf-idf}(t,d) = \mathrm{tf}(t,d) \times \mathrm{idf}(t)$$

其中，occu(t,d) 表示词汇 t 在文档 d 中出现的次数，而 df(t) 则表示包含词汇 t 的文档数，n 为总文档数。

2.3 朴素贝叶斯分类器

贝叶斯分类是一类分类算法的总称，这类算法均以贝叶斯定理为基础，其中朴素贝叶斯分类是其中最简单，也最常见的一种分类方法。可以这样形式化定义一般的分类问题，设未知样本为 $\boldsymbol{x}$，某标签为 c，则将 $\boldsymbol{x}$ 分类为 c 的概率为 $P(c|\boldsymbol{x})$，通过计算对所有 c 的 $P(c|\boldsymbol{x})$，最终将该样本标记为使该概率最大的标签。于是问题就转换为了如何求 $P(c|\boldsymbol{x})$。目前有两种基本策略，分别为判别式模型和生成式模型，前者直接对 $P(c|\boldsymbol{x})$ 建模，主要包括决策树，BP 神经网络，SVM 等（后文将会介绍），后者则首先对联合概率分布 $P(\boldsymbol{x},c)$ 建模，再由概率公式

$$P(c|\boldsymbol{x}) = \frac{P(\boldsymbol{x},c)}{P(\boldsymbol{x})}$$

获得 $P(c|\boldsymbol{x})$，其主要代表为贝叶斯分类器。

根据贝叶斯定理，有

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})}$$

其中 $P(c)$ 为先验概率，是样本空间中各类样本所占比例，可通过统计频率来估计，$P(\boldsymbol{x})$ 为证据因子，与类别无关，另外 $P(\boldsymbol{x}|c)$ 为样本相较于类标记的类条件概率，亦称似然，这是最主要的难点。如果直接估计 $P(\boldsymbol{x}|c)$ 的话，由于 $\boldsymbol{x}$ 是包含多个属性的向量，将可能面临组合爆炸或者样本稀疏的问题。

朴素贝叶斯分类器（下面简称 NB）采用了“属性条件独立性假设”[15]：对已知类别，假设所有属性相互独立。于是公式就变成了

$$P(c|\boldsymbol{x}) = \frac{P(c)P(\boldsymbol{x}|c)}{P(\boldsymbol{x})} = \frac{P(c)}{P(\boldsymbol{x})}\prod_{i=1}^{d} P(x_i|c),$$

由于 $P(\boldsymbol{x})$ 对所有类别都相同，所以最终的标签

$$h_{nb}(\boldsymbol{x}) = \text{argmax}_{c\in C} P(c)\prod_{i=1}^{d} P(x_i|c)$$

设样本集为 D，标签为 c 的样本集为 D_c，标签为 c 且在第 i 个属性上取值为 x_i 的样本集为 D_{c,x_i}，则在离散情况下

$$P(c) = \frac{|D_c|}{|D|}, \quad P(x_i|c) = \frac{|D_{c,x_i}|}{|D_c|}.$$

2.4　支持向量机

在二分类问题中，两个类别之间的差异往往足够大以至于能够在样本空间中找到一个超平面将它们分开。支持向量机（下面简称 SVM）就是一种希望能找到在原问题属性空间上找到最优的划分超平面的模型 [16]。所谓最优，直观上就是指处于两个类别“正中间”的位置，这时它的鲁棒性最好，泛化性也最强。假设超平面方程为 $\boldsymbol{w}^T\boldsymbol{x} + b = 0$，正负类的标签分别为 $y_i = \pm 1$，离超平面距离最近的点（称之为支持向量）刚好使得 $\boldsymbol{w}^T\boldsymbol{x} + b = \pm 1$，则优化问题可表述为

$$\begin{aligned} \text{argmax}_{\boldsymbol{w},b} \quad & \frac{2}{\|\boldsymbol{w}\|} \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1, \quad i = 1, 2, ..., m. \end{aligned}$$

或者

$$\begin{aligned} \text{argmin}_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1, \quad i = 1, 2, ..., m. \end{aligned}$$

这是一个受多个条件限制的最优化问题，可以自然地想到用拉格朗日乘子法构造对偶问题去求解，限于篇幅省略一些细节[1]。首先，引入拉格朗日乘子 $\alpha_i \geq 0$，可以得到拉格朗日函数

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{m} \alpha_i (y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b) - 1),$$

接着令 $L(\boldsymbol{w}, b, \boldsymbol{\alpha})$ 对 $\boldsymbol{w}$ 和 b 的偏导为 0，可得

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0,$$

最后回代得到新的优化目标2.2

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j - \sum_{i=1}^{m} \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1, 2, ..., m. \end{aligned} \tag{2.2}$$

得到所有 α_i 后，最终得到的超平面可表示为

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + b = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i \boldsymbol{x} + b$$

注意到，在求解的过程中，该优化问题需要满足两个条件才可使用对偶性，一个是该问题必须是凸优化问题，一个是它必须满足 KKT 条件[2]，即要求

$$\begin{cases} \alpha_i \geq 0, \\ y_i f(\boldsymbol{x}_i) \geq 1, \\ \alpha_i (y_i f(\boldsymbol{x}_i) - 1) = 0. \end{cases}$$

这意味着当 $y_i f(\boldsymbol{x}_i) > 1$ 时，一定有 $\alpha_i = 0$；换句话说，在训练完成后，所有的非支持向量不必保留，最终模型仅与支持向量有关，这也体现了支持向量机解的稀疏性。

具体应该如何解2.2呢？最常用的方法叫做 SMO，该方法会在每一步中选取两个需要更新的变量 α_i 和 α_j，固定它们以外的所有参数，求解然后更新 α_i 和 α_j，不停执行直至收敛。除此之外有时还需引入软间隔的概念，以允许一部分但尽可能少的样本不满足约束。

[1] https://en.wikipedia.org/wiki/Duality_(optimization)

[2] https://en.wikipedia.org/wiki/Karush-Kuhn-Tucker_conditions

2.5　神经网络

在机器学习中，神经网络一般被定义成由具有适应性的简单单元组成的广泛并行互联的网络, 它的组织能够模拟生物神经系统对真实世界物体所作出的反应。在生物神经系统中，每个神经元与其他神经元相连，当一个神经元的电位超过阈值时，它就会被激活，然后向其他神经元发送化学物质，从而可能引起其他神经元也被激活。基于此，人们提出了神经元模型，它的输入是来自其他 n 个神经元传递过来的信号，通过不同权重的连接进行运算，将处理后的值与神经元的阈值进行比较，最后将激活函数的处理结果输出。理想情况下，激活函数是跃迁函数

$$\operatorname{sgn}(x) = \begin{cases} 1, & \text{if} \quad x \geq 0, \\ 0, & \text{if} \quad x < 0. \end{cases}$$

但是它具有不连续、不光滑等数学上不好的性质，因此通常采用 Sigmoid 函数

$$\operatorname{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

感知机是一种由两层神经元组成的分类器 [17]，输入层接收信号，乘以各自的连接权后将结果传递给输出层，输出层对各个带权信号求和再进行阈值处理后输出，即

$$y = f(\sum_{i=1}^{n} w_i x_i - \theta).$$

因此，对于某一个分类问题，只需要求出各个连接权 w_i 即可。设某个训练样本为 $(\boldsymbol{x}, y)$，若当前感知机的输出为 $\hat{y}$，则学习规则可形式化表述为

$$\Delta w_i = \eta(y - \hat{y})x_i$$

$$w_i \leftarrow w_i + \Delta w_i$$

其中 $\eta \in (0, 1)$ 成为学习率。可以发现，当感知机预测正确时，权重将不发生变化，否则将会朝正确的方向逐渐靠拢。可以证明，在只有两类样本的前提下，若它们线性可分，则该学习算法一定收敛，否则将会发生震荡。对于线性不可分的情况，需要引入新的模型。

多层感知机（或称多层前馈神经网络，以下简称 MLP）就是这样一种模型，它是在普通感知机的基础上增加若干隐层得到的 [18]。在 MLP 中，每层神经元都与下一层神经元全连接，且不存在同层连接和跨层连接。对于复杂的 MLP，之前简

单的学习规则不再适用，于是引入了一个扩展的学习算法，即误差逆传播算法（以下简称 BP）[19]。

直观上，BP 是一个迭代算法，它会根据每一次迭代的结果自后往前更新参数值，这也是其名字的由来。具体而言，假设有一个拥有 d 个输入神经元，q 个隐层神经元和 l 个输出神经元的 MLP，给定训练集 $D = (\boldsymbol{x}_i, \boldsymbol{y}_i)$，其中 $\boldsymbol{x}_i \in R^d, \boldsymbol{y}_i \in R^l$，令隐含层的第 h 个神经元的输出结果为 b_h，则需要训练的参数包括 θ_j（输出层第 j 个神经元阈值），γ_h（隐含层第 h 个神经元阈值），v_{ih}（输入层的第 i 个神经元与隐含层第 h 个神经元之间的连接权重），w_{hj}（隐含层的第 h 个神经元与输出层第 j 个神经元之间的连接权重），再假设对样本 $(\boldsymbol{x}_k, \boldsymbol{y}_k)$，MLP 的实际输出为 $\hat{\boldsymbol{y}}_k$。

在训练的过程中，首先需要进行前向计算。首先计算隐层的输出，

$$\beta_h = \sum_{i=1}^{d} v_{ih} x_i, \quad b_h = f(\beta_h - \gamma_h).$$

其次计算输出层，

$$\hat{\alpha}_j = \sum_{i=1}^{q} w_{hj} b_h, \quad \hat{\boldsymbol{y}}_j^k = f(\alpha_j - \theta_j).$$

然后便可以基于梯度下降策略，以目标的负梯度方向对参数进行调整了。首先需要计算 Δw_{hj}，这也是符合 BP 的字面意义。对于误差 E_k，设学习率为 η，则

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}}$$

而根据链式法则，有

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{\boldsymbol{y}}_j^k} \cdot \frac{\partial \hat{\boldsymbol{y}}_j^k}{\partial \alpha_j} \cdot \frac{\partial \alpha_j}{\partial w_{hj}}$$

再令

$$g_j = -\frac{\partial E_k}{\partial \hat{\boldsymbol{y}}_j^k} \cdot \frac{\partial \hat{\boldsymbol{y}}_j^k}{\partial \alpha_j} = -(\hat{\boldsymbol{y}}_j^k - \boldsymbol{y}_j^k) f'(\alpha_j - \theta_j) = \hat{\boldsymbol{y}}_j^k (1 - \hat{\boldsymbol{y}}_j^k)(\hat{\boldsymbol{y}}_j^k - \boldsymbol{y}_j^k)$$

则有

$$\Delta w_{hj} = \eta b_h g_j$$

类似的还可推导出

$$\Delta \theta_j = -\eta g_j, \quad \Delta v_{ih} = \eta x_i e_h, \quad \Delta \gamma_h = -\eta e_h$$

其中

$$e_h = b_h (1 - b_h) \sum_{j=1}^{h} w_{hj} g_j$$

有了参数的更新公式，BP 算法就可以运行了。见算法2.1。

Algorithm 2.1 Backward Propagation

Input: The train dataset $D = (\boldsymbol{x}_k, \boldsymbol{y}_k)_{k=1}^m$; The learning rate η;

Output: MLP with fixed $w_{hj}, v_{ih}, \theta_j, \gamma_h$;

```
Initialize parameters randomly.
for not converged yet do
    for (x_k, y_k) ∈ D do
        Compute ŷ_k;
        Compute g_j;
        Compute e_h;
        Update w_hj, v_ih, θ_j, γ_h by computing Δw_hj, Δv_ih, Δθ_j, Δγ_h;
    end for
end for
```

2.6　集成学习

根据之前的数学推导可以想见，当样本发生变化时，MLP 受到的影响远远大于前两种分类器，因此可以很自然地想到，有没有可能将多个 MLP 结合起来形成一个性能更加强大的分类器呢？答案当然是肯定的，这就引入了集成学习的概念 [20]。

集成学习大致分为两类，Boosting 和 Bagging，前者主要用在个体学习器存在强依赖关系的情况下，后者则用于弱依赖关系。对于 Boosting，由于个体学习器依赖强，因此只能串行生成，每次根据前一次的误差调整训练数据的分布生成新的学习器，最后将学习器综合起来得到集成学习器 [21]。而 Bagging 则一般会并行化生成所有个体学习器，它们之间的区别就在于获得了不同的训练样本，这些样本主要是通过对原始样本集自主采样得到的 [22]。而 MLP 适用于后者。见算法2.2。

Algorithm 2.2 Bagging of MLP

Input: The train dataset $D = (\boldsymbol{x}_k, y_k)_{k=1}^m$; The base model $\mathcal{L}$; The learning rounds T;

Output: $H(\boldsymbol{x}) = \operatorname{argmax}_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y)$;

```
for t = 1, 2, ..., T do
    h_t = L(D, D_bs);
end for
```

2.7 本章小结

本章介绍了系统用到的分析工具的数学原理，包括布隆过滤器、tf-idf、朴素贝叶斯分类器、支持向量机、神经网络和集成学习。同时，本文并未对对偶问题等较为复杂的问题进行详细的数学推导，有兴趣的读者可参阅其他文献。

第 3 章　舆情分析系统的构建

本章将完整地实现一个简单的热点事件舆情分析系统，包括获取数据、数据预处理、使用分类器训练与评估数据等步骤。

3.1　获取数据

3.1.1　选择爬取对象

显而易见，首先需要找到一个合适的平台来获取舆情。为了使得获取的信息更有效，本文想要的平台应具备以下几种特点：

- 它的用户使用的语言应尽可能统一，否则会需要额外的翻译工作；
- 它的用户量应很大，覆盖面很广，这样更能反映真实的情况；
- 它是一个以文字内容分享为主的网站。

第一点使得 Twitter 被排除在外，虽然它后面两条都符合，同时也是全世界最早的微博平台，但其英文以外的内容已到了不能忽视的程度，这会加大工作难度，于是本文把目光集中在国内网站上。虽然国内交流平台形形色色，但真正谈得上有影响力的却屈指可数，如果再考虑第三点，会发现只有新浪微博较为符合本文的要求。新浪微博是一个基于用户关系的信息分享、传播以及获取信息的平台，它占据中国微博用户总量的 57%，以及中国微博活动总量的 87%，是中国大陆访问量最大的网站之一，截至 2012 年 12 月底，新浪微博注册用户已超 5 亿，日活跃用户数达到 4620 万，用户每日发博量超过 1 亿条[1]。虽然新浪微博现在存在着营销号众多、垃圾广告信息遍布等诸多问题，但无论根据网民的覆盖面还是舆论的总量，它都是目前为止最好的选择。

3.1.2　分析网页结构

新浪微博拥有好几个不同的站点，常用的有 https://weibo.cn 和 https://weibo.com。为了方便起见，本文选择其网页结构较为简单的 https://weibo.cn

[1] https://zh.wikipedia.org/wiki/新浪微博

作为爬取对象。回到本文的需求，即获取某一热点事件的舆情信息，有时候还需要指定相应的时间段，于是需要使用微博的高级搜索功能。使用这一功能时，会发现搜索结果最多显示 100 页，其中每一页都有 10 个结果，在这种条件下，每次最多可以获得 $10 \times 100 = 1000$ 份原始文本。

接下来需要具体分析网页的组织方式，以方便后续处理。经过观察，发现微博文本大致分为以下几种情况：

- 用户 A 的原创内容，有的时候包括相关话题和几组图片。
- 用户 B 单纯转发 A 的原创内容，此时会以特定格式显示原先的微博内容。
- 用户 B 转发并评论 A 的原创内容，此时除了原始的微博内容，还会在原微博下方展示转发理由。
- 用户 B 转发并评论 A 的原创内容，用户 C 又对用户 B 的评论进行转发和评论，以此类推，可能还会出现用户 C，D，最后每个人的评论都会出现在最后转发的用户的微博内容中，格式为“D 的转发理由//@C:C 的转发理由//@B:B 的转发理由//@A:A 的转发理由”。

第一种情况可以直接获取其内容；而对于后三种情况，不仅需要记录原始微博，同时也要把所有相关评论记录下来。

3.1.3　制作和使用爬虫

定位到了信息的位置，接下来本文需要定制一个爬虫来自动获取信息。为了更好地绕过反爬虫机制，需要模拟 http 请求头，其中大部分成分都是固定的，特别需要注意的是 User-Agent（用户代理字符串）和 Cookie（用户身份信息），可以设置一系列对应的值，在每次爬取时随机挑选某种组合，以降低被检测出来的几率，然后利用构造好的头去抓取目标网页的内容，最后利用上一节的知识将网页中的有效信息存储下来。

处理完一页后，如果存在下一页，会继续抓取下去，这时候要注意应及时更新 url 的值并充分考虑可能出现的情况。注意，适当地增加每次页面访问的间隔时间也有助于防止触发反爬虫机制。当爬取完预期的页数或已经爬完所有页面后程序会停止，这时便获得了测试集，即还未被分类的数据。

3.2　数据预处理

在计算机诞生的早期年代，便有“Garbage in, garbage out”的说法，指如果将错误的、无意义的数据输入计算机，则计算机的输出结果也一定是错误的、无意义的，该原则在其他领域同样有效。统计数据表明，数据预处理在真正的数据挖掘任务中通常会占用 70% 左右的工作量。为了使分析结果更加符合事实，有必要对获得的原始数据进行充分的预处理。

3.2.1　使用布隆过滤器去重

观察在上一节中获取到的数据，很容易就能发现其中有很多重复文本。造成重复文本有以下几种原因：

- “水军”小号恶意刷屏，以达到营销目的。
- 官方号大量进驻，其微博内容被转发多次。
- 微博内容太短，与其他微博恰好“撞车”，这种情况较为稀有，可忽略。

第一种情况是最需要防范的情况，第三种不需要处理，而第二种则需要分析。一般来说，官方号发布的内容较为中立客观，含有情绪较少，比较难以归类，如果分类错误，就会造成大量的误判；同时，人们在转发该条微博的时候，一般也会给出自己的态度，完全可以据此而不是通过记录大量重复的原始微博获得有效的信息。因此，本文认为对微博内容进行去重是有好处的。

关于去重，最朴素的做法是将所有文本进行两两比较，该方法的准确率是完美的，对于少量文本而言也是有效的，但对于大量文本来说，该方法的时间复杂度显然是无法容忍的。因此需要考虑效率更高的算法，比如布隆过滤器 (Bloom Filter)。

经过多次重复实验，发现每次爬取时文本数一般不超过 2000，令可接受的假阳性概率为 1%，则根据公式2.1有

$$m = -\frac{2000\ln 0.01}{(\ln 2)^2} \doteq 19170 \doteq 20000,$$

此时最佳 k 值约为 7。

具体应用时，需要首先初始化一个空的布隆过滤器，每读入一条微博，首先需要判断该条微博是否已在集合中，如果没有，就将该条微博加入集合，否则不做任何事。见算法3.1。

Algorithm 3.1 BloomFilter

```
bf = new BloomFilter(n = 2000, p = 0.01);
dinstinct_weibo = new List();
for each element x in weibo do
    if x not in bf then
        bf.add(x);
        dinstinct_weibo.add(x);
    end if
end for
return dinstinct_weibo;
```

3.2.2　对句子进行分词

清理完数据后，需要对每条文本进行情绪预测。传统的方法会侧重于语言学与计算机科学的结合，对每个句子进行语法分析和语义分析等等，最后得到确定的结果，其优势是可解释性强，能够嵌入学科知识，劣势则是准确度较低。而现在主流的方法则是机器学习，主要分为两类，无监督学习和有监督学习，有时候也会结合两者，即半监督学习。无监督学习的目标主要是聚类，即为每个样本找到对应的簇。而在有监督学习中，主要是要对样本进行分类，即为每一个样本找到对应的标签。如何获得标签信息呢？这就引入了训练集的概念。对于有监督学习，输入数据大致分为两个部分——训练集和测试集，其中训练集也可继续分出一部分作为验证集。训练集是拥有标签的数据，其功能是生成一个模型（称之为分类器），依靠这个模型才能对测试集（无标签的数据）进行标签的预测。现在已经拥有了测试集，即上一节中清洗好的无标签的数据，还需要训练集，从哪儿获得呢？幸运的是，互联网上有不少已经标注好的开源微博文本，本文暂时以拥有十万行标注文本的数据集[2]作为训练集。

一般来说是不能直接将整句话作为有监督学习中分类器的输入的，因为计算机并不具备理解句子的能力，必须将其转换为更加基本的单位才行。对于新浪微博中的主流语言，也就是中文来说，这中间还有一个必要的步骤——分词。

与英文句子不同，中文句子字与字、词与词之间没有间隔，且现代汉语的基本表意单元是词，而英文句子每个单词直接以空格间隔，且大多数自成表意单元。基于这点，需要特定的工具将中文句子分割成词，这样才能得到有意义的结果。若

[2]https://github.com/SophonPlus/ChineseNlpCorpus/tree/master/datasets/weibo_senti_100k

自己实现该功能，则需要对中文语料库进行训练，代价庞大，因此本文选择了目前最流行的中文分词库 jieba 协助完成这一目标。

3.2.3　删除无意义词汇

无论是中文句子还是英文句子，它们往往都包含冗余甚至无意义的词汇。对于英文来说，the,a,that,which 信息量很低；对于中文来说，“的”，“了”，“总之”这类助词在语义层面也可忽略不计。除此之外，常见的标点符号或者无意义的字符序列也应去除。如果不去除这类词汇，不仅会导致更长的训练时间开销，而且可能会由于被常用词支配或是被无意义词妨碍而产生不理想的结果，因此只有去除它们，才能达到更好的效果。有两种方法可以去除不想要的词汇。

- 利用正则表达式。正则表达式可以很好地处理大多数模式匹配的问题，因此用其删除具有一定模式的词汇，比如数字等。很容易解释数字是无意义的，因为它们常常表示日期或价格等，而这与态度很难关联。
- 利用停词表。对于不具备特定模式的词汇，比如标点符号或是“的”,“了”等，可以将其收录于一张本地的停词表中，将其交给具有停词表接口（比如 TfidfVectorizer）的函数或是手动匹配以进行无意义词汇的删除。

3.2.4　将分词结果转换为向量

大多数分类器也不会接受分词后的句子作为输入，因此需要继续将分词结果转换成具有相等长度且元素为实数的向量。有两种方法可供选择。

利用 TfidfVectorizer

sklearn 提供了 tf-idf 的实现接口，即 TfidfVectorizer，需要注意的是，其默认参数为识别两个单词以上组成的词汇，这是针对英文的做法，对于中文，则需要将其调整为识别一个词以上。TfidfVectorizer 会计算一个文档中的所有词汇的个数作为生成向量的属性维度。对于某个句子中未出现而在其他句子中出现的词汇，其在该句中生成的 tf-idf 向量中对应位置的值为 0（因为它的 tf 值为 0）。由于文档中的绝大多数词汇都不会在某一个句子中出现，因此整体构成了巨大的稀疏矩阵（即绝大多数位为 0），应按字典序存储，以减少存储开支。

图 3.1: Word2vec 举例

利用 Word2vec

Word2vec 是用来产生词向量的神经网络模型，它将分词后的句子群作为输入，最终将对应位置的词汇转换为长度统一的向量。图3.1展示了一个英文句子的其中四个单词通过 Word2vec 训练后的结果，

其前四个维度的值分别代表了该单词在某一维度上的相关性。在实际情况中，并非总能解释每个维度的意义，但却通常可以利用词向量的余弦相似度来得到两个词的相关性。为了结果能够处理，需要令每个句子以相同维度的向量形式出现，因此采取算术平均的方式处理每一个句子中的所有词向量。这种做法的缺陷是无法很好地解释每个句子向量的具体含义，但是效果却很好。相比于 tf-idf 向量，句子向量维数较少，且非零值很少，因此整体上构成了规模较小的稠密矩阵。

3.3　分类器的训练与评估

预处理过后，需要使用分类器对训练集进行训练和评估。下面讨论不同的分类器的选择及其原因。

1. 朴素贝叶斯分类器

   首先考虑朴素贝叶斯分类器。经过观察，“属性条件独立性假设”对于 tf-idf 矩阵基本成立，因为其属性是词汇的 tf-idf 值，而任意两个词汇的 tf-idf 值不相关。而对于词向量来说，这条假设就不成立，因为词向量是根据上下文得出的，与文本距离较近的词向量相关。基于以上理由，本文认为 NB 在输入是 tf-idf 矩阵时效果更好，实验结果也表明该分析是基本正确。为了使 NB 获得更为有效的输入，还需要降低句子的属性维度并且使用拉普拉斯修正来避免概率为 0 的情况。

关于句子的降维，可以利用 k 近邻学习或主成分分析等多种降维算法实现，但方便起见，也可直接选择设置 TfidfVectorizer 里的 max_features 参数，或者混合使用 sklearn 的 feature_selection 库中的 SelectKBest 方法和 chi2 方法（卡法检验）以达到同样的效果。实验结果表明，合适的降维除了能加快训练速度外，也能提升分类准确率。表3.1为 NB 在不同最大属性个数下的表现 (卡方检验，alpha=1700)：

表 3.1: 朴素贝叶斯分类器的 F1 分数与最大属性数的关系 (alpha=1700)

Max Feature	F1-score
30	0.92653
100	0.94267
300	0.94789
500	0.94670
1000	0.94491

当最大属性个数在 300 左右时，NB 在验证集上取得了较好的效果。

由于“属性条件独立性假设”，NB 忽视了词语之间的顺序关系，即在 NB 眼里，“我要吃饭”和“饭要吃我”是相同的意思，称之为“词袋”。事实上，除了 NB，生成式模型还包括半朴素贝叶斯分类器和贝叶斯网等等，它们复杂度更高，也考虑了属性之间的依赖关系，但在文本分类问题上却不见得比 NB 效果更好，这也是本文选择 NB 的原因之一。

2. 支持向量机

其次考虑支持向量机。容易发现，当输入为 tf-idf 矩阵时，属性空间的维度较大，冗余度高，易于区分不同的实例，因此线性核 SVM 便能够解决。但当输入为词向量矩阵时，由于是低维度稠密矩阵，划分的效果往往不佳，于是本文选择 tf-idf 矩阵而不是词向量矩阵作为 SVM 的输入。

相较于 NB，SVM 不需要假设属性之间相互独立，泛用性更强；相比于之后的神经网络，SVM 具有坚实的数学理论，能够得到全局最优解。表3.2展示了不同最大属性个数下 SVM 的表现 (卡方检验)：

表 3.2: 支持向量机的 F1 分数与最大属性数的关系

Max Feature	F1-score
100	0.95511
300	0.97451
500	0.97439
700	0.97417
1000	0.97393

当最大属性个数在 300 左右时，分类效果最好。

3. 神经网络

最后考虑神经网络，分为单个 MLP 和多个 MLP。

(1) 单个 MLP

由于 MLP 需要为每一个属性赋予权值，而当属性值为 0 时，权值则无意义，因此应当减少属性值为 0 的情况，而稀疏矩阵显然不符合要求。另一方面，根据前面的分析可知，训练一个复杂度较高的 MLP 的开销是比较大的，而低维向量相较于高维向量具有训练时间上的优势。综合以上两方面，本文选择词向量平均后的句子向量构成的矩阵作为输入。

为了更好地权衡训练开销与预测性能，同样需要仔细设计 MLP 的结构。本次实验中的 MLP 结构如图3.2所示。

该 MLP 为双隐层感知机，每一层的神经元个数为 100。由于该网络具有较强的表示能力，容易遭遇过拟合，因此本文在训练的过程中需要采取“早停”的策略，使训练在验证集误差开始升高时终止。为了使训练更快，可以在开始训练时将学习率保持在较高的水平，当训练快要结束时，再适当地调低学习率，以便更精确地定位局部最优点。表3.3展示了当输入为不同维度的句子向量时 MLP 的表现 (LR= 0.001)：

在句向量维度超过 200 后，验证集分数差距就很小了。

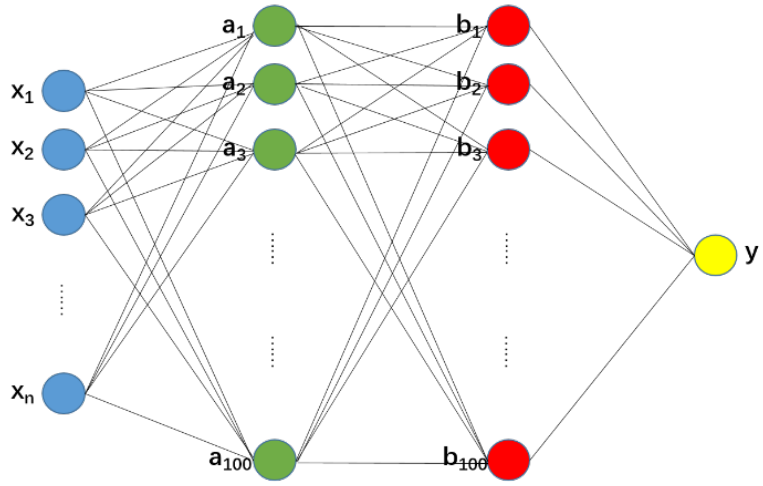

图 3.2: MLP 结构

表 3.3: 多层感知机的验证集分数与词向量维度的关系 (LR=0.001)

Dimension	score
50	0.90316
100	0.94425
200	0.95283
300	0.95750
500	0.95200

(2) 以 MLP 为基学习器的集成分类器

由于 MLP 对于训练样本相当敏感，因此可以通过数据样本扰动生成多组训练数据，然后交给具有同样参数的 MLP 基学习器进行训练；在预测阶段，对于某一个测试样本，选取其中预测次数最多的种类作为结果即可。除了神经网络，“Bagging”对未剪枝的决策树等输入敏感型的学习算法相也当有效。令句向量维度为 100，用上节的 MLP 作为基学习器进行集成。表3.4展示了不同个数的 MLP 集成的表现：

表 3.4: 集成分类器的验证集分数与基学习器个数的关系 (Dimension=100)

Number	score
3	0.95916
5	0.96091
10	0.96133

随着基学习器个数的增加，集成效果也越来越好。

3.4　系统实现

本系统采用 python 语言实现，主体分为代码文件、数据文件和模型文件。其中，代码文件分为三个模块，分别为爬虫模块、预处理模块和分类器模块，各模块的功能已在前文介绍过；数据文件则又分为原始数据文件、中间数据文件和结果数据文件，其中原始数据文件包括未原始训练集、爬取到的测试集以及停词表，中间数据文件包括去重后的测试集以及保存下来的用来加速代码运行速度的 tf-idf 文件和 Word2vec 文件，最后的结果文件则包括对每一个测试文本的预测标签。模型文件则为训练好的分类器。

下面介绍各个代码模块的设计框架。

3.4.1　crawl.py

该部分主要包括五个方法，分别是 set_first_url，set_next_url，downloadHtml，parse_content 和 crawl_pages。

方法 set_first_url(keyword, start_time, end_time, rand)

顾名思义，该方法设置了爬取对象的第一个 url 值，该 url 值主要需要设置四个参数，分别是关键字、起始时间、终止时间和随机值，其中随机值需要不定时更新。

方法 set_next_url(keyword, start_time, end_time, page_number)

与上一个方法类似，区别在于该方法设置的是下一页的 url 值，因此需要传入页号。

方法 downloadHtml(url)

该方法需要传入一个 url 值，然后根据该值将对应的网页内容完整地下载下来。为了做到这点，需要模拟一个 http 请求头 header，该数据结构包括多个字段，其中尤其需要注意的字段为“user_agents”和“cookies”，前者与用户浏览器相关，后者则包括了用户的登录信息，填错将导致爬取失败。然后将 url 值和 header 值传

给 urllib.request 包里的 Request 函数以构造网页访问请求，再用该包中的 urlopen 函数协助获取网页信息。

方法 parse_content(html_str)

该方法会对一个完整的 html 网页进行解析，然后将需要的信息保存到 list 中。首先需要利用 bs4 库中的 soup 函数对网页内容进行结构化重组，找到所有的 div 块，根据在 3.1.2 节中获得的知识对每个 div 块中的可能存在的微博文本进行解析。具体而言，本文发现大部分微博文本都是以“:”开始的，并且“[组图]”或者“赞[..]”后面的内容一般没有意义，因此可以以此为依据对内容进行抓取。其他情况也可以用类似的想法处理。

方法 crawl_pages(keyword, start_time, end_time, rand, page_number)

该方法调用了上述四种方法，先构造一个首页的 url，对该网页下载并解析后，再迭代地对后续网页进行类似的操作。最后需要把 list 中的结果写入到测试集文件中去。

3.4.2　preproc.py

该部分包括四个方法，分别是 clean_data，tfidf_list，word2vec_list 和 ret_data。

方法 clean_data()

该方法分别对训练集和测试集进行了数据清洗。对于训练集，该过程为删除无意义词汇和分词，而对于测试集，则多了去重的步骤。由于使用的去重算法是布隆过滤器，因此引入了 bloom_filter 库中的 BloomFilter 对象来协助工作，算法流程见3.1。

方法 tfidf_list(write)

该方法能将清洗过的数据转换成 tf-idf 矩阵，用到的外部库为 sklearn.feature_extraction.text，本文采用该库中的 TfidfVectorizer 对象作为向量化工具。对于训练集，调用该对象的 fit_transform 方法生成 tf-idf 矩阵，对于测试集则直接调用 transform 方法。write 为布尔值，当它为真时，需要完整地生成一次 tf-idf 矩阵并且保存到本地文件中，否则直接从文件中读取 tf-idf 矩阵。

方法 wrod2vec_list(write)

该方法能将清洗过的数据转换成词向量矩阵，用到的外部库为 gensim.models，本文采用该库中的 Word2Vec 对象作为向量化工具。与上个方法不同，需要将训练集和测试集合并后才能训练词向量，同时对于每一句话的每个词向量，进一步去计算它们在所有维度中的算术平均值作为该词向量的表征，以此来简化后续处理。write 的功能与上个方法类似。

方法 ret_data(write, select)

该方法调用了以上两种方法，write 的功能已提到过，而 select 则是用来选择生成 tf-idf 矩阵或是词向量矩阵。该方法返回训练集矩阵、训练集标签和测试集矩阵。

3.4.3　clf.py

该部分包括六个方法，分别是 solve_by_clf，solve_by_NB，solve_by_SVM，solve_by_MPL，solve_by_Bagging_MPL 和 predict_with_batch。

方法 solve_by_clf(clf, X_train, y_train)

该方法的功能仅仅是调用传入的分类器的 fit 方法，然后返回训练好的分类器。这层封装的主要意义在于将各方法之间逻辑上的关联显性表示出来。

方法 solve_by_NB(X_train, y_train)

该方法将会使用 sklearn.naive_bayes 库中的朴素贝叶斯模型。该库一共有三种 NB，分别是 GaussianNB，MultinomialNB 和 BernoulliNB，其中 GaussianNB 的先验为高斯分布，MultinomialNB 的先验为多项式分布，而 BernoulliNB 的先验为伯努利分布。一般来说，如果样本特征的分布大部分是连续值，使用 GaussianNB 会比较好；如果样本特征的分大部分是多元离散值，使用 MultinomialNB 比较合适；而如果样本特征是二元离散值或者很稀疏的多元离散值，应该使用 BernoulliNB。除此之外，sklearn 的官方文档还提到，在实践中，tf-idf 等小数形式的计数值也适合使用 MultinomialNB 进行分类[3]，这也是本文选择它的原因。MultinomialNB 的主要

[3]https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#sklearn.naive_bayes.MultinomialNB

参数为光滑系数 alpha，在 MAX_FEATURE = 300 的条件下对不同的 alpha 值进行实验。结果如表3.5所示。

表 3.5: 朴素贝叶斯的验证集分数与 alpha 值的关系 (Max feature=300)

alpha	score
1	0.92906
10	0.92979
100	0.93603
1000	0.94709
1700	0.94881
2000	0.94806
3000	0.94785

不难发现，当 alpha 为 1700 时，效果最好。

方法 solve_by_SVM(X_train, y_train)

该方法将会使用 sklearn.svm 库中的支持向量机模型。该库有三种 SVM，分别是 SVC，NuSVC 和 LinearSVC，其中 SVC 是基于 C 语言库 libsvm 实现的，其复杂度较高，一般无法扩展到 10000 各样本的数据集；NuSVC 与 SVC 类似，但多了一个用来控制支持向量个数的参数 nu；LinearSVC 是线性核的 SVC，但实现方式基于 liblinear，因此它对大规模的数据集更加适用，这也是本文选择它的原因。当涉及到多分类问题时，LinearSVC 采用的默认策略是 OvR，但剩下的策略并不包括 OvO，如果想要使用 OvO 的话，还需要调用 sklearn.multiclass 中的 OneVsOneClassifier 对 LinearSVC 进行封装，但这又带来一个新的问题，即在验证或预测阶段很容易由于模型太多而超内存，因此要定制一个分批预测的函数。但由于本文讨论的是二分类问题，因此不细究。

方法 solve_by_MPL(X_train, y_train)

该方法会使用 sklearn.neural_network 中的 MLPClassifier 对象，为了防止过拟合，本文将其设置为较为简单的双隐层结构，同时将学习率调为 0.001，学习方式

设置为“adaptive”，即一开始学习率较大，在即将收敛时减小学习率，早停策略设置为真，最大循环次数设置为 200，每次训练的 batch 大小设置为 256。

方法 solve_by_Bagging_MPL(X_train, y_train)

该方法会使用上面的 MLPClassifier 对象和 sklearn.ensemble 库中的 BaggingClassifier 对象。基学习器 MLPClassifier 的参数设置与上面相同，再将 BaggingClassifier 的基学习器个数设置为 10 即可。

3.5　实验及结果分析

引言提到了新型冠状肺炎，于是本文便以“肺炎”作为关键词进行实验。从 3 月 1 日起，以周为单位从新浪微博上抓取数据，用上一节中训练的 MLP 的 Bagging 集成作为分类器进行预测。结果如图3.3所示。

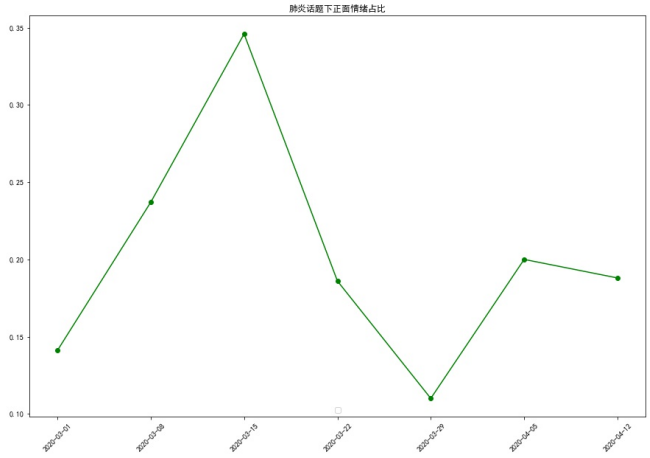

图 3.3: 3.1-4.12 内肺炎话题情绪占比

该结果显示，人们在“肺炎”话题下的讨论的正面情绪最高占比仅为 35%，这也符合预期，因为传染病对人们造成的负面影响远大于正面影响。结果还显示，3 月 15 日到 3 月 21 日这段时间群众正面的情绪到达高点，这与当时国内确诊人数不断下降而全球疫情尚处于初级阶段的情况相符合，而之后的负面情绪的迅速爬升，则可以用全球疫情形势紧张以及海外输入病例开始攀升来解释。

与之相对的是，如果用 SVM 来进行训练和预测文本的话，会发现结果与 MLP 相差甚远——它会把大多数的样本的情感倾向预测为正面，这显然是与预期不符

的。联想到之前是用 tf-idf 值作为 SVM 的输入的，该值不考虑句子所在的上下文，再加上训练样本与测试样本在性质上存在一定差别（虽然它们都是微博文本，但早年的微博风格相比现在已变化很多了），因此可能导致 SVM 过拟合了训练数据，使得预测准确度大大下降，这也是为什么如今大家更加青睐 Word2vec 与神经网络的组合的原因。

3.6　本章小结

本章介绍了分析系统构建的全流程和实验结果，包括数据的获取、数据预处理、用不同的分类器进行训练和评估、具体的代码框架等。除了描述结果外，本章将更多的精力放在了对每一步为何如此的阐述上。

第 4 章　结束语

4.1　论文的主要工作

本文针对日趋重要的网络舆情构建了一个分析系统。首先是从新浪微博上爬取大量相关话题下的文本，再对这些文本进行去重处理，再针对微博特有的结构进行分析并过滤无关信息，接着将文本转换成实数矩阵，最后利用 sklearn 中的各种分类器训练标注好的文本，并对未标注的文本暗含的情绪进行预测。实验结果表明该分析系统是有效的。

4.2　进一步的工作

今后的工作主要集中在各个模块的并行化上，真正实现对大数据的即时、快速、增量、高效处理。除此之外，本文还需进一步提高系统的复杂性、鲁棒性和可扩展性，使之能处理多个平台上的信息源，对结果的预测更加精确。

参考文献

[1] 许鑫, 章成志. 互联网舆情分析及应用研究 [J]. 情报科学, 2008, (8): 1194-1200.

[2] Alasadi, Suad A and Bhaya, Wesam S (2017). ”Review of data preprocessing techniques in data mining”. Journal of Engineering and Applied Sciences. 12 (16): 4102–4107.

[3] Goodwin, Bob; Hopcroft, Michael; Luu, Dan; Clemmer, Alex; Curmei, Mihaela; Elnikety, Sameh; Yuxiong, He (2017). ”BitFunnel: Revisiting Signatures for Search”. SIGIR: 605–614.

[4] Yakunin, Alex (2010-03-25). ”Alex Yakunin’s blog: Nice Bloom filter application”.

[5] ”Issue 10896048: Transition safe browsing from bloom filter to prefix set. - Code Review”.

[6] Xue, Nianwen and Converse, Susan P. (2002), ”Combining Classifiers for Chinese Word Segmentation”. COLING-02: The First SIGHAN Workshop on Chinese Language Processing.

[7] Luhn, Hans Peter (1957). ”A Statistical Approach to Mechanized Encoding and Searching of Literary Information”. IBM Journal of Research and Development. 1 (4): 309–317.

[8] Mikolov, Tomas; et al. (2013). ”Efficient Estimation of Word Representations in Vector Space”.

[9] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina (11 October 2018). ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”.

[10] Tsou B K Y, Yuen R W M, Kwong O Y, et al. (2005), Polarity classification of celebrity coverage in the Chinese press[C]. Proc of International Conference on Intelligence Analysis.

[11] Pang B, Lee L, Vaithyanathan S. (2002), Thumbs up?: Sentiment classification using machine learning techniques[C]. Proc of the ACL Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics.

[12] Taboada M, Brooke J, Tofiloski M, et al. (2011) Lexicon-based methods for sentiment analysis[J]. Computational Linguistics, 2011, 37(2): 267-307.

[13] Bloom, Burton H. (1970), ”Space/Time Trade-offs in Hash Coding with Allowable Errors”, Communications of the ACM, 13 (7): 422–426.

[14] Rajaraman, A.; Ullman, J.D. (2011). ”Data Mining”. Mining of Massive Datasets. pp. 1–17.

[15] Maron, M. E. (1961). ”Automatic Indexing: An Experimental Inquiry” . Journal of the ACM. 8 (3): 404–417.

[16] Cortes, Corinna; Vapnik, Vladimir N. (1995). ”Support-vector networks” (PDF). Machine Learning. 20 (3): 273–297.

[17] Freund, Y.; Schapire, R. E. (1999). ”Large margin classification using the perceptron algorithm” (PDF). Machine Learning. 37 (3): 277–296.

[18] Hastie, Trevor. Tibshirani, Robert. Friedman, Jerome. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, NY, 2009.

[19] Rumelhart, David E.; Hinton, Geoffrey E.; Williams, Ronald J. (8 October 1986a). ”Learning representations by back-propagating errors”. Nature. 323 (6088): 533–536.

[20] Polikar, R. (2006). ”Ensemble based systems in decision making”. IEEE Circuits and Systems Magazine. 6 (3): 21–45.

[21] Zhou Zhi-Hua (2012). Ensemble Methods: Foundations and Algorithms. Chapman and Hall/CRC. p. 23.

[22] Breiman, Leo (September 1994). ”Bagging Predictors” (PDF). Department of Statistics, University of California Berkeley. Technical Report No. 421.

致 谢

感谢陈家骏教授在选题和设计阶段的指导意见以及对我工作学习方面的关心。

感谢陈钦霖、陈昕元、陈亚栋三位同学兼舍友在整个过程中给予的协助。

感谢南京大学和各位老师的培养，感谢四年以来与我共同学习进步的同学们。

感谢张希对我的督促，没有她，我将很难完成这件事。

最后要感谢我的家人和朋友们，你们的支持对我来说是千金也不换的财富。