分类号_____ 密级 _____

UDC _____

学 位 论 文

基于深度学习与自然语言处理的金融市场走势预测

作 者 姓 名 ：

指 导 教 师 ：

东北大学信息科学与工程学院

申请学位级别： 硕士 学科类别： 专业学位

学科专业名称： 控制工程

论文提交日期： 论文答辩日期：

学位授予日期： 答辩委员会主席：

评 阅 人 ：

东 北 大 学

2018 年 11 月

A Thesis in Control Engineering

Financial Market Trend Forecast Based on Deep Learning and Natural Language Processing

By

Supervisor:

Northeastern University

November 2018

独创性声明

本人声明，所呈交的学位论文是在导师的指导下完成的。论文中取得的研究成果除加以标注和致谢的地方外，不包含其他人已经发表或撰写过的研究成果，也不包括本人为获得其他学位而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文作者签名：

日　　期：

学位论文版权使用授权书

本学位论文作者和指导教师完全了解东北大学有关保留、使用学位论文的规定：即学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人同意东北大学可以将学位论文的全部或部分内容编入有关数据库进行检索、交流。

作者和导师同意网上交流的时间为作者获得学位后：

半年 □　　一年□　　一年半□　　两年□

学位论文作者签名：　　导师签名：

签字日期：　　签字日期：

摘 要

随着国内金融市场的不断发展，参与金融市场投资的投资者也在逐年增加。投资者参与金融市场的主要方法为股票交易，但由于股票价格具有较强的波动性，如果投资者无法准确把握股价的走势，不但无法获取收益，反而会蒙受经济损失。另外，当市场遭受大型波动时，准确地把握市场走势也有利于主管机构进行干预与颁布救市政策。

针对以上应用场景，本文提出一种基于深度学习与自然语言处理技术的市场走势预测系统，可以根据金融舆论文本与股票数值数据对市场中个股的涨跌趋势做出正确率较高的预测。本文的主要工作如下：

首先，提出了一种改进的词向量二阶段训练方法。传统词向量模型在情感分析领域的弊端，导致了最终的预测模型难以区分舆论文本中的乐观倾向词语与悲观倾向词语。为了缓解这一情况，本文借鉴了文本情感分类中所采用的方法，并结合本文应用背景对该方法进行了改进，实验表明采用改进后方法调整得到的模型增加了词向量包含的情感信息。

其次，提出了一个对本文中的金融舆论文本进行表征的神经网络模型。由于需要对大规模文本数据直接处理，所以本文基于分层模型的思想，提出了一种适用于舆论文本数据的网络模型，该模型称为基于自注意力机制的双向 LSTM 与 CNN 分层模型。与其他模型的对比试验结果表明，该模型兼顾了训练耗时与模型精度。另外，本文还提出了一种对自注意力机制的改进，实验表明改进后的模型的精度有所提升。

最后，提出了一个基于舆论文本数据与股票数值数据的混合模型，并与数据采集模块共同构成一个预测系统。该模型同时处理文本数据与数值数据，并在最终一部分添加了堆叠式循环神经网络与高速公路网络结构，可处理多日时长跨度的数据，该模型称为基于改进自注意力机制的双向 LSTM 与 CNN 分层混合模型。最终的实验结果表明，本文提出的模型可在 5 日时长的数据上达到 59.2%的二分类正确率。

关键词：走势预测；神经网络；自然语言处理；词向量；自注意力机制

Abstract

With the continuous development of domestic financial market, the number of investors participating in financial market investment is increasing year by year. The main method for investors to participate in the financial market is to buy and sell stocks. However, as the stock price is highly volatile, if the investors cannot accurately grasp the trend of the stock price, they will not get profits, but will suffer economic losses. In addition, when the market suffers large fluctuations, accurate control of the market trend is also conducive to the authorities to intervene and promulgate rescue policies.

In view of the above application scenarios, this thesis proposes a market trend prediction system based on deep learning and natural language processing technology, which can predict the rise and fall of individual stocks with high accuracy according to the text of financial public opinion and stock numerical data. The main work of this thesis is as follows:

First, an improved method of word vector quadratic training is proposed. The disadvantages of the traditional word vector model in the field of emotion analysis lead to the difficulty in distinguishing the optimistic words from the pessimistic words in the public opinion. In order to alleviate this situation, this thesis used the method of text emotion classification for reference, and combined with the background of this thesis to improve the method, experiments showed that the improved method adjusted the model added the emotional information contained in the word vector.

Secondly, a neural network model is proposed to characterize the text of financial public opinion. Due to the need of direct processing of large-scale text data, this thesis, based on the idea of hierarchical model, proposes a network model suitable for public opinion text data, which is called Bi-LSTM and CNN hierarchical model based on self-attention mechanism. The results of comparison with other models show that the model takes into account the training time and precision. In addition, an improved self-attention mechanism is proposed in this thesis, and experiments show that the accuracy of the improved model is improved.

Finally, a mixed model based on public opinion text data and stock numerical data is proposed. This model processes both text data and numerical data at the same time, and adds the stacked recurrent neural network and highway network structure in the final part, which can process data with a multi-day span. This model is called Bi-LSTM and CNN layered hybrid

model based on improved self-attention mechanism. The final experimental results show that the model proposed in this thesis can reach 59.2% accuracy in the data of 5 days.

Key words: trend forecast; neural network; natural language processing; word vectors; self-attention mechanism

目 录

第 1 章 绪 论

1.1 课题研究背景及意义

1.1.1 课题研究背景

近年来，随着国内经济的不断发展，国内金融市场的活跃度与受关注度也大幅提升，其中股票市场作为金融市场的主要组成部分，其运行情况可以在一定程度上反应国内或某地区的经济走势，以及某个行业的发展趋势。股票市场的运行状况在经济发展中的作用不容忽视。

在近几年内，国内股市发生过几次较大的波动，其中最著名的一次发生在 2015 年，从当年上半年开始中国股市进入“牛市”，大量个股持续性上涨，直至当年六月份，上证指数由年初的 3200 点上涨至 5187 点，涨幅为 61.8%；而后，股市由“牛市”突然转变“熊市”，波及范围之大被投资者称为“千股跌停”，上证指数下探至 2850，跌幅 45.2%，不少投资者遭受财产损失。自股灾发生之后一段时间，主管部门接连发布救市政策进行干预，股票市场才得以稳定下来。若能在一定程度上提前预知市场走势，则有助于预防市场出现此类大波动，并且主管部门能更早采取措施进行干预。

与其他国外股票市场参与者由机构投资者构成不同，中国股市的参与者主体目前仍然是个体投资者。与机构投资者相比，个体投资者没有接受过专业投资训练，专业知识不足，其决策依据主要为社会舆论以及近期内的股票价格，通常无法准确地把握市场动向，容易追涨杀跌，频繁买卖。因此，相比与国外股票市场，国内股票市场的换手率一直居高不下，从而造成了股市的频繁震荡。如果能有一套模型指导个体投资者进行投资的话，既能减少个体投资者的资产损失亦或扭亏为盈，也有利于稳定国内股票市场的走势。

随着信息技术的不断进步，互联网的影响范围与使用者的人数也屡创新高。截止 2018 年 6 月，中国互联网使用人数达 8.02 亿，互联网普及率达 57.7%[1]。在信息时代，由于社交平台的兴起，每个人既是信息的接收者也是信息的创造者，人们在社交平台上表达自己观点时，观点同时也会影响到其他人。在针对股市的社交平台上，比如雪球网与东方财富网股吧等平台，股票投资者们经常对股市走势、经济政策与新闻等信息发表自己的见解，并与其他投资者相互交流相互影响。因此，社交网站上的舆论对个体投资者的决策产生了重要影响，进而影响了股票市场的走势。

传统的股票预测模型主要采用以基于经济指标与数值数据的机器学习模型，由于数据匮乏以及模型表征能力不强，导致预测模型难以达到很好的表现。近年来，在数据量高速增长的推动下，深度学习技术与数据科学发展迅猛，基于深度学习技术的新方法在各个行业与领域掀起了一阵狂潮。在股票预测领域，深度学习技术为其带来了一轮新的发展机遇与发展动力，本文基于深度学习与自然语言处理技术，尝试直接对大规模文本进行建模，进而对股票乃至整个股市的走向作出预测。

1.1.2 课题研究意义

本课题的研究意义主要分为两个部分，即理论意义与实用价值。本文的理论意义在于，由于早期的股票预测模型主要采用时序建模以及机器学习等方法，首先模型学习能力较弱，其次其仅采用了股市价格数据，部分模型采用了股票基本面指标，数据较为匮乏。近年来部分学者将文本数据融入预测模型中，其中针对舆论文本的研究大多数采用情感指标法，即不将文本纳入学习模型中，而是采用一定的数值指标代表舆论文本特征，这种方法无法利用文本数据蕴含的意义。本课题基于自然语言处理技术对大量舆论文本数据直接处理，并与深度学习技术融合进行建模得到预测模型。

本课题的实用价值在于，一个准确且快速的金融市场走势预测模型不仅可以用于研究股票走势与其他信息的相关性与因果关系，比如股价、公司经营状况、舆论信息等，也有利于辅助金融市场主管部门发布市场政策以及对市场进行必要的监管与干预。另外，走势预测模型也可以帮助个人投资者避免资金亏损以及实现小额盈利。

1.2 国内外研究现状

1.2.1 传统预测模型与机器学习模型

股票走势预测问题最早起源于上世纪的国外研究，目前广泛应用的股票预测技术主要可以被分为三类：投资分析方法，时间序列方法以及机器学习方法，以下对三个方法进行介绍。

投资分析法是指基于金融学与经济学的人工预测方法，主要通过股市某时段内的走势，股票基本信息与金融指标，以及公司财务状况等数据，基于决策者的主观理解与直觉来进行人工决策。这种方法具有一定的宏观预测能力，但在短期的市场运行中则表现较差，另外由于决策者的直觉千差万别，使得本方法稳定性与可靠性较低。

时间序列方法认为，股市的运行曲线有一定的规律性，所以依靠历史走势可以得到将来的走势，所以时间序列方法尝试对股票运行曲线直接建模。该方法的经典算法为

ARIMA 模型，中文名为自回归积分滑动平均模型(Autoregressive Integrated Moving Average Model,简称 ARIMA)，由 Box 等人[2]提出，其大体思想为，将被预测的时间序列视为一个随机序列，并且其可以采用某个模型进行近似描述，一旦获得该模型之后，就可以基于时间序列的过去值预测未来值。时间序列方法适合解决较为简单的线性问题，但对于股市运行规律时间序列模型表现较差。

由于机器学习技术的发展，产生了许多股票预测的新方法。机器学习方法在股票预测中的应用最早可追溯于上世纪 90 年代，最早由 White[3]采用 BP 神经网络预测 IBM 的每日回报率，受限于当时的计算能力，该模型未能达到较为准确的水平。Kimoto 等人[4]采用神经网络模型开发了“TOPIX”系统并在日本股市中进行了预测。Rodriguez 等人[5]建立了基于 ARIMA 与 BP 神经网路的混合模型，并在西班牙股市中进行了预测，其结果显示在模拟交易中提出的模型所获的利润远高于其他模型。郑建刚[6]等人将改进的遗传算法与 BP 神经网络相结合，得到了一个预测模型。Kim 等人[7]采用支持向量回归（SVR）模型对股票收盘价格进行预测，模型的精度高于 BP 神经网络。王波等人[8]比较了神经网络模型与 ARIMA 模型，结果显示神经网络模型表现较好。秦焱等人[9]将粒子群优化算法与 BP 神经网络相结合，神经网络模型的精度有所提升。Sui 等人[10]将小波去噪与支持向量回归模型相结合进行回归预测，结果证明相比于单纯的支持向量回归模型，小波去噪可以提升模型精度。Zhang 等人[11]研究了股票价格走势的形状，提出了一种基于形状的特征，并结合其他数值特征采用随机森林进行预测。吕涛等人[12-13]采用 K 线序列的形态与位置的相似性建立预测模型，获得了较好的结果。

1.2.2 采用文本数据的预测模型

随着自然语言处理技术的发展，部分学者开始研究如何将新闻文本及舆论文本与机器学习方法相结合来对金融市场走势作出预测。对于新闻事件文本数据的研究，Ding 等人[14-16]采用新闻事件作为数据，采用开放信息提取技术（Open IE），结合知识图谱技术对新闻中的对象与行为进行抽取并向量化，并采用神经张量网络学习对事件的向量化表示，最终与全连接神经网络与卷积神经网络结合对股票走势进行预判，在对标普 500 中的个股的预测表现上达到了较高的准确率。Akita 等人[17]证明了新闻标题可以有效替代新闻的内容，于是采用词向量将新闻标题文本转化为向量特征与股票的数值类特征合，并基于循环神经网络对股票收盘价进行回归预测，在针对日本股市的模拟交易中获得了极高的收益率。Hu 等人[18]提出了一种新型的网络结构，称之为混合注意力网络，该模型直接利用新闻文本进行预测股票第二日的走势，最终在 A 股市场中的实验表明其在三类分类预测的情况下准确率可达 50%。

对于金融市场的舆论文本数据的研究，张凯等人[19]与董理等人[20]均采用将每日舆论文本数据转化为特定的数值指标，并分别基于长短期记忆神经网络与 SVR 进行了预测。Guo 等人[21]研究并采集了雪球网的用户评论，并分析社交舆论中的情感指数与股票市场走势之间的关系，其结果表明，当舆论中的情绪指数较高时，舆论数据可以用来预测股市。Zhang 等人[22]研究表明雪球网中的舆论文本的数量与股市中当天的换手率存在相关关系，并且研究了雪球网的用户结构并计算了用户的影响力值，最终基于文本情感指标与用户转发关系与影响力构成的有向图构建了一个二类分类预测模型。Zhang[23]在另一项研究中通过对投资者情绪与新闻事件以及股价的波动构造了一个三维张量，并采用张量分解对股票走势进行预测，模型中还考虑了不同股票之间的走势关系。Huang 等人[24]基于 Zhang 的研究，在张量分解的基础上加入了长短期记忆神经网络，相比之前的模型表现有所提升。Das 等人[25]采用推特数据流与历史股价数据构建了一个实时的股票预测系统，该系统达到了较高的精度。

1.3 本文主要内容与结构安排

1.3.1 本文主要研究内容

本文主要采用深度学习与自然语言处理技术，基于前几日的数据预测某只股票的第二日股票价格走势。预测任务为一个二分类任务，若第二日的收盘价高于前一日的收盘价，则被划分为上涨类，若第二日收盘价等于或低于前一日的收盘价，则被划分为下跌类。预测过程所采用的的数据有两个来源，分别为金融舆论文本数据以及股票运行数值特征数据。本文中的金融舆论文本的来源为采用爬虫系统爬取的雪球网的文本数据，其中包括投资者讨论、公司新闻与公告发布、以及投资机构的研究报告；本文的股票数值数据包括股票的价格指标与流通情况指标等数据。通过对以上两类数据采用预测模型进行同时处理，得到第二天股票走势的预测结果。针对以上过程，本文的主要研究内容有三部分并分别如下。

首先，本文研究了对金融领域情感倾向词语的二阶段训练。如果要采用神经网络模型处理自然语言文本，除去数据预处理过程之外，第一步应建立文本的词向量表示模型。本文在采用前人所提出的词向量模型对预处理后的舆论文本数据进行训练得到词向量表示之后，发现词向量模型无法对带有情绪与情感倾向的词语进行准确区分。由于本文需要预测模型能够从海量的舆论文本数据中识别出投资者们的情绪倾向，比如对股市的乐观态度等情绪，如果词向量模型无法对带有情感倾向的词语进行准确表示，则预测模型会受到严重的影响。针对以上问题，本文对词向量的二阶段训练进行了研究，以改善

情感倾向词词向量表示的质量。

其次，本文研究了如何采用神经网络对大规模舆论文本直接进行建模，并将文本数据整合至预测模型中。由于本文获取到的舆论文本数据众多，传统机器学习方法难以对其进行建模；而深度学习技术善于处理大量的非结构化数据，所以本文选取神经网络模型作为文本处理模型。本文基于深度学习技术，研究了如何在保证模型运行速度的情况下同时处理大量文本数据，以及如何尽可能地提升模型的表现。

最后，结合已得到的文本神经网络模型，本文研究了如何将文本数据与股票数值数据相结合给出最终的预测结果，并使最终的预测模型拥有处理多日数据的能力。在得到文本神经网络模型之后，本文设计了新的模型用于处理数值数据，以及将两类数据进行合并。另外，当模型可以处理多日数据时，相比仅采用前一天的数据进行决策，模型能获得更多的决策数据，从而提升预测表现。

1.3.2 本文结构安排

本文主要研究基于深度学习与自然语言处理技术的市场走势预测，本文的详细内容在每一章的结构安排如下：

第 1 章为本文的绪论，主要阐述了本文课题的研究背景与研究意义；总结了国内外本领域常见研究方法、研究的历史进程与近期研究状况，并指出了前人研究中存在的不足之处；阐述了本文的主要内容与其他各章节的安排。

第 2 章主要阐述课题中所使用的数据的相关问题，主要介绍了两类不同数据的数据类型、数据获取、数据存储等问题；对采集到的文本数据进行了数据分析，获得了文本数据的主要统计特征；对获取到的数值数据进行皮尔逊相关性分析，并依据特征重要程度舍弃了部分重要度较低的特征；最终对文本数据进行清洗并分词之后，将其存储为特定格式的训练集，以便在之后的章节中直接使用。

第 3 章主要内容为金融文本词向量模型，首先简介了词向量的概念，论述了经典词向量模型 Word2Vec 与 GloVe；其次，采用收集到的文本数据，基于上文的两个模型，选定合适的参数并训练了两个不同的词向量表，并依据评价标准选取了 Word2Vec 作为最终采用的词向量模型；最后，针对词向量模型在情感分析领域的缺陷，本文对前人的研究工作进行改进后，得到了一个适用于金融领域的无监督词向量二阶段训练算法，并且采用本算法调整后的词向量减轻了调整前存在的缺陷。

第 4 章主要内容为确立了金融文本的深度学习模型，首先简介了神经网络的基本结构，并且对文本模型中采用的主要结构，如文本卷积神经网络与自注意力的双向长短期记忆神经网络进行了详细介绍；其次，基于分层注意力网络设计本文的文本模型，由于

模型速度速度较慢，所以基于其分层建模的思想，本文提出了一种基于卷积神经网络与长短期记忆神经网络的混合模型，实验证明本文提出的模型兼顾了训练速度与精度；最后，本文还对自注意力机制进行了改进，实验证明改进后的模型精度有所提高。

第 5 章主要内容为混合预测模型的设计以及实验结果分析，首先给出了整个市场走势预测系统的结构，并设计了金融数值数据部分的处理模型，与第四章得到的文本模型相结合，获得了最终的混合预测模型；其次介绍了文本实验部分的若干细节，包括损失函数以及评价标准，另外简介了在训练过程中使用的一些技巧；最终本章展示了训练过程，并在关键变量维度上以及不同模型的表现上对实验结果进行了分析。

第 6 章统领全文，对本文的工作进行总结，并分析了文本研究工作中的一些独特之处，以及展望后续的研究方向。

第 2 章 金融数据的获取与分析

在建立走势预测模型之前，需要先获取相应的数据，同时进行数据预处理等操作。本章主要介绍文本中所涉及的数据的相关问题，包括数据类型、数据获取与存储等操作，并进行了相应的统计分析与相关性分析，最后根据预测模型的需要将数据存储为特定格式的训练集。

2.1 数据类型

2.1.1 舆论文本数据简介

已有研究表明，社会舆论会对股票走势产生影响[14-16]。为了获得相比于仅使用金融数值类数据的预测模型，本文采集了舆论文本数据作为数值类数据的扩充数据。在前人研究工作中，舆论文本数据的获取来源主要是东方财富网的股吧与新浪财经，而本文采用的数据来源为雪球网。雪球网是一个专业的投资社交平台，用户可以在平台上发表意见或分享投资组合，相比于东方财富网与新浪财经，雪球网有着更多的移动端用户，平台影响力比前两者更大，所以本文选择舆论文本数据的来源为雪球网。

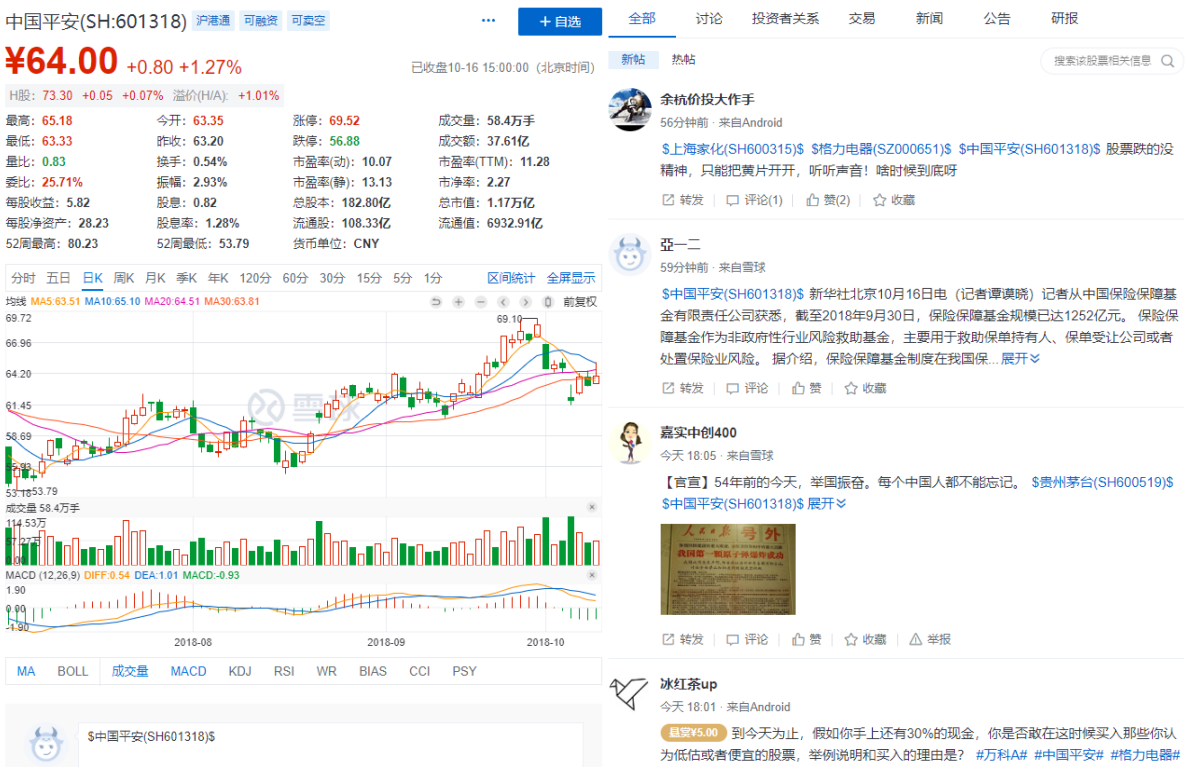

图 2.1 雪球网股票界面

Fig. 2.1 Stock interface of Xueqiu

图 2.1 展示了雪球网“中国平安”(SH601318)股票的界面，左侧为股票基本信息及 K 线图等信息，右侧为关于本只股票的舆论信息，分为六部分，其中“讨论”与“交易”栏目由用户发布，“投资者关系”栏目由企业发布，“新闻”，“公告”，“研报”由雪球网整理并发布。本文获取除“投资者关系”栏目以外的所有其他栏目的文本数据。在保存文本数据的同时，也会保存本条文本数据的相关数据，比如点赞数、回复数、转发数等，以及发布用户的粉丝数及发文数等，以便在模型中使用。

2.1.2 数值类数据简介

为了得到较为精确的预测结果，预测模型需要股票的某些数值特征。本文所采用的数值数据来源为 Tushare 接口包。Tushare 是一个开源 Python 财经数据接口包，主要提供沪深股票的日线行情、复权因子、停复牌信息、每日基本面指标，以及公司财务数据与交易市场数据等。本文的预测模型主要使用的数据为日线行情，也就是股票每日的价格数据，比如开盘价收盘价等；以及每日基本面指标，代表着股票流通性的一些特征。图 2.2 展示了 Tushare 提供的主要接口。

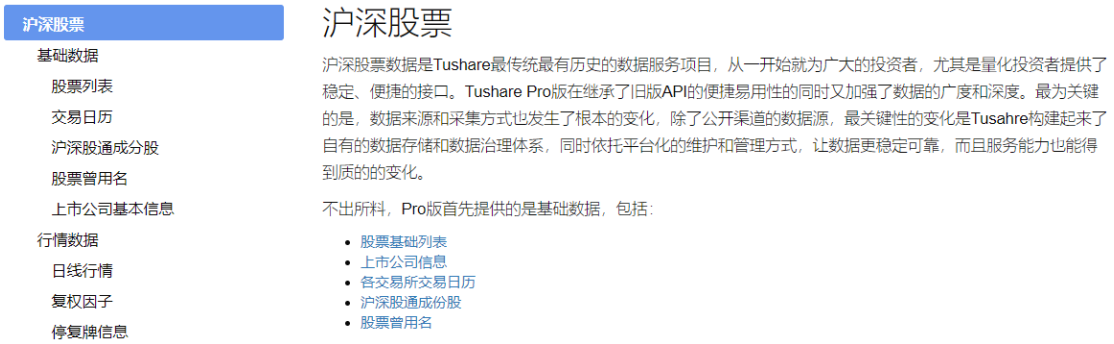

图 2.2 Tushare 主要接口

Fig. 2.2 The main APIs of Tushare

2.2 数据获取与存储

2.2.1 舆论文本数据采集

由于雪球网没有提供数据获取接口，所以只能通过网络爬虫来进行采集其数据。本文主要采用了基于 Python 的 Scrapy 爬虫框架与 Beautiful Soup 工具包。Scrapy 是一个开源爬虫框架，主要优点有速度快，流程化，数据结构化等。Beautiful Soup 是一个可以从 HTML 或 XML 文件中提取文本数据的 Python 工具包，在本文中其用途主要为采集某些 Scrapy 无法采集，但可以正常显示在前端界面上的信息。

Scrapy 框架主要流程如下：引擎从调度器中取出一个 URL 并将其封装为 Request 传

递给下载器，下载器下载页面并封装为一个 Response 传递给爬虫。爬虫对 Response 解析，若得到实体，则将实体传递给管道，管道一般与数据库相连；若解析出 URL，则将其放入调度器的优先队列中排队。全部完成后引擎继续从调度器中取出 URL 继续上述步骤。Scrapy 框架模块的主要功能由表 2.1 描述。

表 2.1 Scrapy 模块主要功能
Table 2.1 The main function of Scrapy's modules

英文名	中文名	主要功能
Scrapy Engine	引擎	框架核心，处理整个系统的数据流
Scheduler	调度器	接受 URL 请求并压入优先队列
Downloader	下载器	下载网页内容并返回给爬虫
Spiders	爬虫	从网页中提取链接或所需信息
Pipeline	管道	处理爬虫从网页中抽取的实体并持久化
Downloader Middlewares	下载器中间件	处理引擎与下载器之间的请求与响应
Spider Middlewares	爬虫中间件	处理爬虫的响应输入与请求输出
Scheduler Middewares	调度器中间件	处理引擎与调度器之间的请求与响应

2.2.2 舆论文本数据存储

从互联网上采集到文本数据后，需要将其存入数据库以便后期进行数据处理。本文采用 MongoDB 数据库存储文本数据。MongoDB 是一种非关系型数据库，操作简单，支持丰富的查询表达式。文本数据的部分列在数据库中存储的格式如表 2.2 所示。

表 2.2 文本数据的存储格式
Table 2.2 The storage format of text data

标签名称	中文名称	数据类型	例子
id	编号	int32	105233572
symbol	股票代码	string	“SH600585”
title	标题	string	“对海螺水泥 2017 年报的几点印象”
comment	文本内容	string	“看看今天外资是否会是净买入，有多少？”
like_count	点赞数	int32	5
created_at	创建时间	int64	1523848644000
user_id	用户编号	int64	8006757304

文本数据在数据库中存储示例如图 2.3 所示，其中由于条目的项目过多，所以展示

图中只选取了部分主要项目。

_id	id	token_class	symbol	title	comment
5adc215b9c54...	1051645	1	SH600585	需求旺盛，上海地...	4月13日起上海地区明珠海螺、金山南方等主要地产...
5adc21419c54...	1051698	1	SH600585	血亏的锦龙股份	有小伙伴，让我集中。其实我集中过，16年分散投资...
5adc20ab9c54...	1038958	1	SH600111	稀土永磁概念股活...	今日早盘，稀土永磁概念股活跃走强，截止发稿，五...
5adc21419c54...	1051794	1	SH600585	真正的成长股，简...	1.净利润涨幅，9年增长10倍以上2.净资产市盈率不...
5adc21559c54...	1052148	1	SH600585	目前看好的10只A...	今根据目前个股基本面的进步明朗，结合估值和...
5adc21559c54...	1052132	1	SH600585	电话会议纪要：从...	@今日话题 $中国神华(SH601088)$ $海螺水泥(SH6...
5adc21439c54...	1051914	1	SH600585	现在的投资组合：...	2018年将投资组合作了进一步调整，主要是：1. 进...
5adc213d9c54...	1051925	1	SH600585	本周天山销售价格...	本周天山销售价格上涨7.1元，西北水泥指数上涨3.0...
5adc21559c54...	1052111	1	SH600585	战争与和平（041	Wx公众号：一石双击周末在两大消息的轰炸下度过...
5adc213d9c54...	1052022	1	SH600585	对海螺水泥2017...	一、行业地位继续提升。2017年，由于固定资产投...
5adc21379c54...	1052280	1	SH600585	大资金买这么多，...	3月26日以来，大盘一直横盘震荡，几乎没涨但这些...
5adc213d9c54...	1052032	1	SH600585	多一点耐心	$恒银金融(SH603106)$ $海螺水泥(SH600585)$ $...
5adc213d9c54...	1051960	1	SH600585	外资周报（4月6...	一周外资增持金额前50名个股 @今日话题海螺水泥...
5adc213d9c54...	1051959	1	SH600585	周初大盘和个股会...	上周A股呈现先扬后抑格局。沪深主板市场在周二和...
5adc213e9c54...	1052881	1	SH600531	今日总结	错过了卫士通，成飞和同仁堂买的还是不错，豫光金...
5adc21419c54...	1051696	1	SH600585	中国建材的持股市	周末没事在家统计下中国建材的持股市值情况，不足...
5adc21559c54...	1052068	1	SH600585	两大新闻，抓不住...	周五晚间，美英法联军合起来欺负叙利亚，给全球股...
5adc21419c54...	1051777	1	SH600585	【继续股票征集】	【继续股票征集】1.业绩确定性高，在行业具有领导...
5adc213d9c54...	1052046	1	SH600585	【外资追踪】哪里...	外资通过"沪股通""深股通"买入A股股票。去年10月1...
5adc213a9c54...	1052150	1	SH600585	4月16日：以涨为主	全国水泥价格继续上涨，以新疆为例，北疆涨完南疆...

图 2.3 文本数据在数据库中的格式

Fig. 2.3 The format of text data in the database

2.2.3 金融数值数据获取

由于金融数值数据可以即用即取，所以无需放入数据库中。具体调用方式为在程序中使用特定的代码申请某种所需的数据，并传入一定参数，如股票代码与时间区间，服务器再以列表形式返回所需数据。图 2.4 采用 Python 集成开发环境 Spyder 的数据浏览功能，展示了日线行情的部分返回数据格式。每日基本面指标返回数据与此类似，所以不再额外进行展示。

a - DataFrame

Index	ts_code	trade_date	open	high	low	close	pre_close	change
62	000056.SZ	20180503	9.99	10.1	9.84	10	9.99	0.01
63	000056.SZ	20180502	10.2	10.2	9.94	9.99	10.2	-0.17
64	000056.SZ	20180427	10.3	10.3	10	10.2	10.2	-0.04
65	000056.SZ	20180426	10.5	10.6	9.99	10.2	10.5	-0.27
66	000056.SZ	20180425	10.4	10.6	10.3	10.5	10.5	-0.01
67	000056.SZ	20180424	10.5	10.6	10.4	10.5	10.5	-0.02
68	000056.SZ	20180423	10.6	10.6	10.4	10.5	10.7	-0.24
69	000056.SZ	20180420	10.7	10.8	10.3	10.7	10.9	-0.12
70	000056.SZ	20180419	10.9	11	10.5	10.9	11	-0.17
71	000056.SZ	20180418	11.5	11.5	11	11	11.5	-0.46
72	000056.SZ	20180417	10.8	11.6	10.7	11.5	11.3	0.2
73	000056.SZ	20180416	10.8	11.3	10.8	11.3	11.9	-0.65
74	000056.SZ	20180413	11.9	11.9	11.9	11.9	13.3	-1.33

Format Resize Background color Column min/max OK Cancel

图 2.4 “日线行情”接口返回值

Fig. 2.4 The return value of “Daily Market” API

2.3 金融舆论文本数据统计分析

本节主要对文本数据进行统计分析，统计分析的结果有助于分析金融市场的受关注度随时间的变化，以及一些公司股票的受关注度异同。本节主要包括文本数据类别分布、文本长度分布，以及文本数量在公司维度的分布以及时间维度的分布。

本文共获取了 11006310 条文本数据，采集自沪深两市 A 股所有公司股票共计 3265 个，时间跨度为 2018 年 4 月 1 日至 2018 年 7 月 31 日。文本数据分为五类，分别为“网友讨论”、“交易信息”、“公司新闻”、“公司公告”、“研究报告”，其分布如表 2.3 所示。由于“交易信息”部分文本数据所含语义太少，例如“我刚刚关注了中国平安，当前价格为 24.5 元”，此类数据不适宜作为训练数据的一部分。除第二类以外，其余数据均在模型中有所使用。

表 2.3 五类数据的分布

Table 2.3 Distribution of five types of data

网友讨论	交易信息	公司新闻	公司公告	研究报告
5088640(46.2%)	2923320(26.6%)	942023(8.5%)	1943558(17.7%)	110827(1%)

图 2.5 为短文本与长文本长度分度的直方图，其中横坐标为文本长度，纵坐标为各个长度区间的频数。短文本与长文本的区分条件是，抓取到的文本是否包含标题项，若不包含标题则是短文本，即推文；若包含标题则是长文本，即文章。统计文本长度分布有助于在第四章中选择合适的神经网络容量参数。由直方图可以看出，短文本数量明显多于长文本数量。短文本长度多数分布在 10 到 150 区间内，长文本多数分布在 120 到 180 区间内。

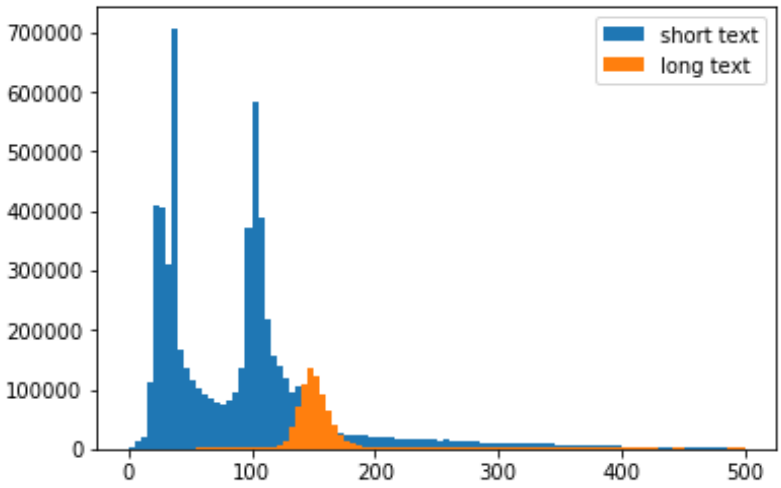

图 2.5 短文本与长文本的长度分布

Fig. 2.5 Length distribution of short texts and long texts

图 2.6 展示了文本数量在公司维度的分布。图中横坐标为文本数量区间，纵坐标为在此文本数量上的公司的频数。由图可以得到，多数公司的文本数量分布在 1800 至 3000 之间。统计文本数量在公司维度的分布可以区分不同受关注度的公司，进一步可以对不同组公司分别构建预测模型。

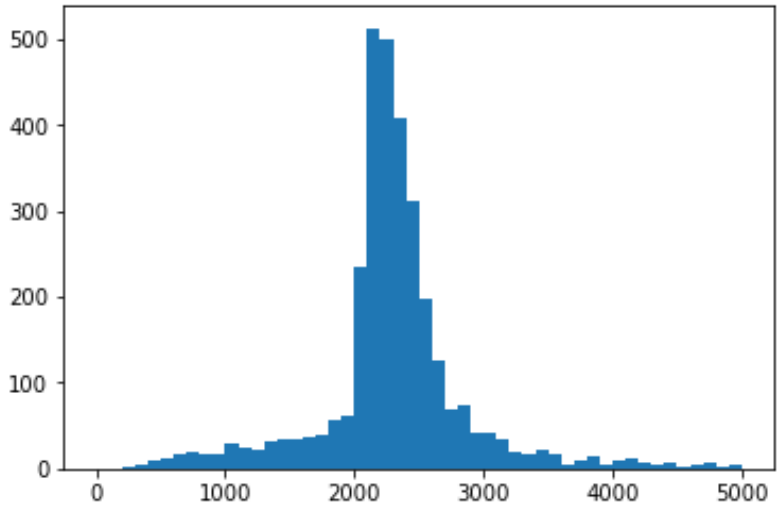

图 2.6 文本数量在公司维度的分布

Fig. 2.6 The distribution of text quantity in the company dimension

图 2.7 展示了文本数量在时间维度的分布。图中的横坐标为日期，时间跨度为 4 月 1 日至 7 月 31 日，纵坐标为当天所有文本量的总和。由于采集最终日期的数据时，爬虫系统发生了一些问题，所以导致最后四天左右的文本数量偏离了正常值。统计文本数量在时间维度的分布，可以帮助理解金融市场总体的受关注度。

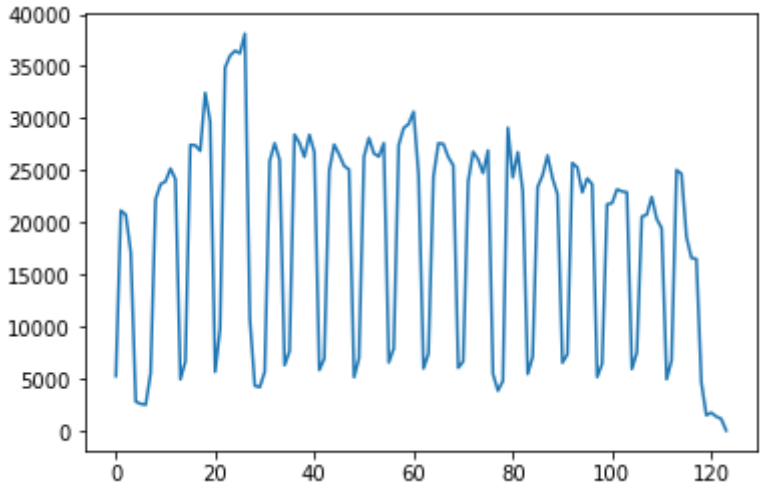

图 2.7 文本数量在时间维度上的分布

Fig. 2.7 The distribution of text quantity in time dimension

从图中可以看出，文本数在约 4 月 23 日至 4 月 27 日达到高峰，此时段为“资管新规”发布前的一星期，可以看到舆论对其反响较大。另外，文本数量随日期变化而呈现

一定的周期性波动性，图 2.8 展示了所有文本数据在周区间上的分布，如图所示，周一至周五的文本总量基本持平，而由于周末闭市的缘故，导致舆论讨论热度有所下降。

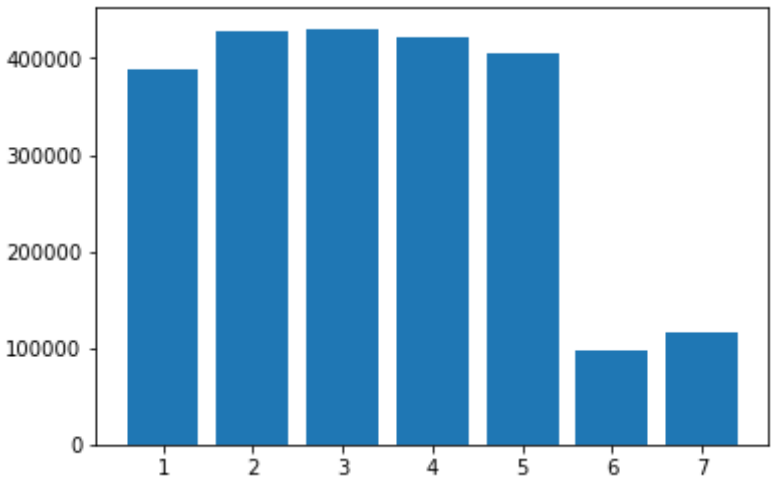

图 2.8 文本数量在周区间上的分布

Fig. 2.8 The distribution of text quantity on weekly interval

2.4 金融数值数据分析

2.4.1 数值特征相关性分析

本文在研究中主要采用 Tushare 工具包中所提供的日线行情与每日基本面指标两个接口所提供的数值类数据，前者提供股票的价格信息与历史走势信息，而后者可以提供一些股票的流通性特征以及公司特征。Tushare 工具包还提供了其他类型的数据，比如公司的财务数据，包括利润表、资产负债表、业绩报告等数据，这一类数据虽然十分齐全，但财务数据主要是公司经营状况的表征，并且财务数据维度极高，而由于其基于公司披露数据进行更新，所以财务数据在长期时间内不会发生改变。由于以上几个原因，文本未将财务数据纳入预测模型的输入数据中。

日线行情接口返回 9 类数据，除去无效数据与重复数据，如股票代码以及日期、涨跌额等数据后，剩余有效数据为 6 类，其分别为：开盘价（open），最高价（high），最低价（low），收盘价（close），成交总手（volume），成交总额（amount）。每日基本面指标接口返回 16 类数据，除去无效数据与重复数据后，剩余 13 类。由于对数据检查后发现其中四类数据会出现空值情况，所以非空数据总共有 9 类，分别为：换手率（turnover rate），自由流通股换手率（turnover rate free），市销率（ps），近十二个月市盈率（ps ttm），总股本（total share），流通股本（float share），自由流通股本（free share），总市值（total mv），流通市值（circ mv）。将两部分数据合并，可以得到 15 类个股每日数值特征，其

数据类型均为浮点类型。

将上文提到的 15 类特征与第二天股票的涨跌组成一个简易数据集，若第二天下跌则标签为 0，若第二天上涨则标签为 1。本节将通过对此简易数据集进行分析，获得特征间相关性信息，并尝试对数值特征进行降维。数值特征的采集范围为沪深 A 股市场，个股数量共计 3265 个，数据的时间跨度选取了与文本数据时间跨度相同，即 2018 年 4 月 1 日至 2018 年 7 月 31 日。按照上文的数据生成方式，除去闭市与股票停牌的日期，共计生成了 258134 条数据，数据特征为 15 维，标签类别为 2。对数据类别进行统计分析，有 139191 条数据类别为 0，占比约为 53.92%，118943 条数据为 1，占比约为 46.08%，也就是说在四个月的时间内，对所有 A 股股票总计有约 54%的交易日股票下跌。

在得到以上数据之后，接下来对数据的特征与类别维度做线性相关性分析。首先对数据的不同列之间计算出皮尔逊相关系数并生成相关系数方阵，该方阵为对称矩阵，方阵中的值$r_{m.n}$代表第m个特征与第n个特征之间的线性相关性，取值范围为$[-1,1]$，大于 0 时两者呈正相关，小于 0 时两者呈负相关，值越大代表线性相关程度越高。热力图（HeatMap）可以用来表示相关系数矩阵，如图 2.9 所示，其中深红色区域代表着高线性相关性，如左上角与右下角，所以由此可以得出结论即数据中含有冗余特征。另外，由最后一行可以得出，“换手率”与“自由流通股换手率”两个特征与涨跌趋势有一定的线性相关性，并且两者相互线性相关。

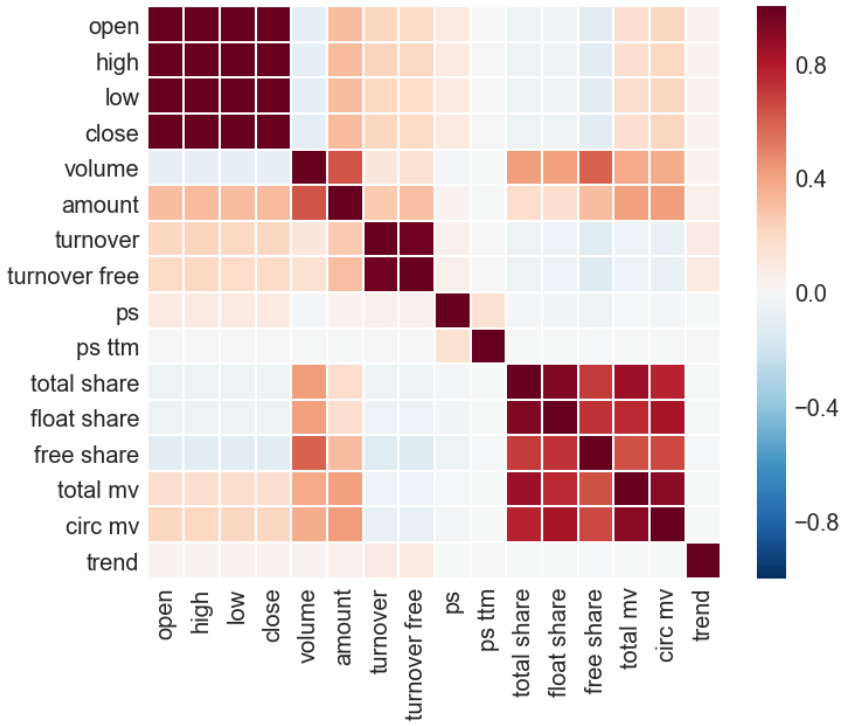

图 2.9 特征相关性热力图

Fig. 2.9 Characteristic correlation HeatMap

2.4.2 数值特征降维

由于特征之间仍然可能存在非线性相关性，所以不能仅依靠皮尔逊相关系数进行特征降维。在第三步中，本文巧妙地引入一种基于梯度提升树模型的特征筛选方式，该方法基于 LightGBM 工具包实现。LightGBM（轻量级梯度提升器）是由微软公司开发的一款基于梯度提升树模型的机器学习工具包，其具有速度快，精度高等优点。由于 LightGBM 的基本模型为决策树，决策树对样本进行学习时，会选取样本中的某一特征作为分离点对训练集切分，并不断循环本步骤直至达到停止条件。基于这个特性，LightGBM 在工具包中加入了一项功能，即记录在训练过程中模型采用各个特征进行分离的样本的个数，采用某一个特征切分的样本数量越多，该特征相对于模型的预测就更重要。

本文基于上述功能，采用 LightGBM 对数据集进行训练并得到特征的重要性如图 2.10 所示，由图中结果可以得出，“total share”与“float share”两个特征对模型的贡献较小，可以去除，将两特征去除后模型准确率由 55.54%上升至 55.60%，由此可以得到降维后特征有助有模型表现的提升。此处的预测准确率仅用于讨论特征的选取，并不属于最终的预测模型的一部分。

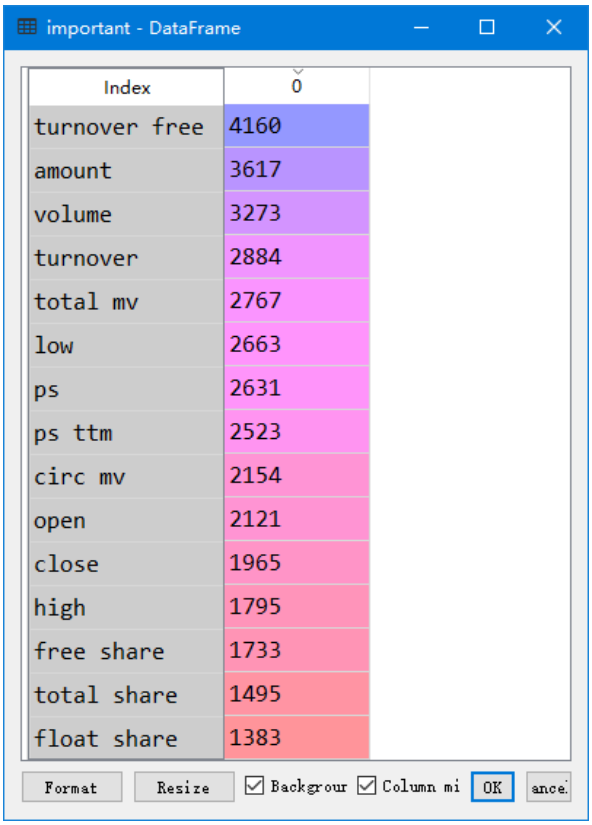

图 2.10 数值特征的重要性排序

Fig. 2.10 The order of importance of numerical features

2.5 金融舆论文本训练集构建

本小节主要内容为对文本进行数据清洗与分词工作，并将处理过后的文本数据与数值数据结合生成预测模型可直接读取的数据集文件，以便于后续章节直接使用数据集进行模型训练。

2.5.1 中文分词

分词是指将一个字符串切分为若干子字符串，并且子字符串需满足独立成词条件的过程。由于英语、法语、西班牙语等语言单词之间采用空格链接，所以依照空格进行分离就可以达到分词效果；而汉语、日语、韩语等语言在句子中没有采用分隔符号对词语进行分词，所以在计算语言学中，对以上语言的文本进行处理首先要采用分词算法进行分词。

常见的分词算法可分为两大类，两类方法分别为无监督算法与有监督算法。无监督算法需要包括词频的补充查询词典，主要方法包括最大匹配法，最少词法，基于有向无环图的最大概率法等；有监督算法需要有标注的数据集，其将分词问题转化为一个序列分类问题，主要采用深度学习模型，如包含条件随机场的双向循环神经网络模型。本文采用基于 Python 的分词工具包 jieba，其采用的算法为基于有向无环图的最大概率法，并采用隐马尔科夫模型自动识别新词，另外还可设置自定义分词规则。

2.5.2 舆论文本数据清洗

通常在网络上直接采集的文本数据会有一些噪声，比如特殊符号、网址、无意义字段、表情、图片等。本文在数据爬取阶段已经通过正则式匹配将非文本数据过滤，如表情与图片等，所以本部分主要对文本噪声进行清洗，清洗过程的方法主要是人工编写过滤规则。

另外，在自然语言处理中，句子中的某些数词与时间状语在多数情况下与句子所表达的意义没有相关性，在这种情况下应该将这一类词替换为同一个词，以避免这些词变为数据噪声从而影响模型表现。例如，在完成数据清洗与分词步骤之后，将“2018 年 7 月 23 日”、“2016 年 9 月”、“15 日”一类的子字符串替换为字符串符号“_DATE_”；将“14:55”、“6 点 15”一类的子字符串替换为字符串符号“_TIME_”；将“十五”、“13”一类的子字符串替换为字符串符号“_NUMBER_”。从训练模型的角度来看，将子字符串替换为上述字符串符号不会改变句意。表 2.4 展示了一些包含噪声的例子以及处理后的结果。

表 2.4 文本数据噪声及过滤规则

Table 2.4 Text data noise and filtering rules

原文片段	处理后（已分词）
回复@那一剑:海螺水泥股价最近不好过。	海螺 水泥 股价 最近 不好 过
$新奥股份(SH600803)$ 净利润 4 个亿？	净利润 _NUMBER_ 个 亿 ？
公司公告下载：http://static.sse.....	公司 公告 下载
7 月 24 日，大盘低开高走	_DATE_ ， 大盘 低开高走
#海螺水泥# 佩服主力的不坚定的筹码！	佩服 主力 的 不 坚定 的 筹码 ！

2.5.3 训练集构建

在经过文本分词与文本数据清洗过后，就可以进行训练集构建了。构建训练集的主要目的为，在训练过程中模型可以直接读取已经构建好的训练数据，而无需进行数据处理等前期工作，加快模型训练速度。本文会以不同的时间长度构建三种不同的训练集，并在第五章中比较它们的表现。

在对文本长度进行统计分析之后，本文选取将每条文本的长度限定于 128 个词，包含标点符号。若一条长文本的长度大于 128，则其将会被切分为多段；若文本长度远小于 128，则将多个长度相似的文本拼凑至大于 128 的 80%，即约 102 个词，并且满足小于 128；若文本长度刚好介于 102 与 128 之间，则不作任何处理直接使用。最终，对于长度小于 128 的文本，为了便于后续处理，需要进行补 0 操作，使所有文本长度相同。

本文在每个自然日内选取 64 个文本作为训练数据的构成，若当天某只股票的文本数大于 64，则将采用热度对文本做降序排序，选取热度值较高的 64 个文本，热度值的计算为点赞数、转发数、回复数、收藏数、打赏数的总和；若当天文本数小于 64，则向文本组补充 128 维度的 0 向量；若当天没有文本，则为了避免向模型中引入噪声，包含该股票该天文本的数据均会被废弃。

对于数值数据，依照上一节的实验结果，本文选择除“流通股本”与“自由流通股本”以外的 13 类特征代表数值数据。结合上文提到的文本组，就可以得到一组当日某股票的全部数据，文本将其称之为“数据单元”。最终数据的组成如图 2.12 所示，一个数据中包含t个数据单元，根据t值的不同，文本制作了 3 个数据集，$t \in \{1,3,5\}$，t取 1 代表采用预测时刻的前一天的数据单元，t取 3 代表采用预测时刻的前三天的数据单元，以此类推；数据中最后一位代表预测时刻的股票走势。三种数据集所含数据数量分别为：$t = 1$时为 203742 条，$t = 3$时为 198549 条，$t = 5$时为 175329 条，三种数据集的类别分布差别不大，上涨类样本占比约为 50.1%至 50.6%，下跌类样本占比约为 49.4%至 49.9%。

造成上涨占比反而比下跌多的原因为，上一步中某些不含文本数据的样本被抛弃，其中较多的样本为下跌类样本。

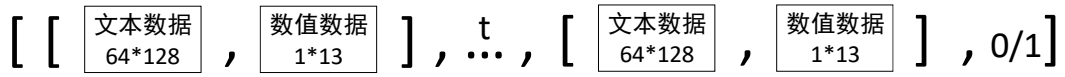

图 2.11 训练集格式

Fig. 2.11 The format of training set

2.6 本章小结

本章主要对课题中所涉及的数据问题进行了阐述，课题中采用了两种数据，即金融舆论文本数据与股票市场数值数据，本章介绍了以上两类数据的数据类型，以及数据获取与存储方法等问题。另外，由于采集到的文本数据量较大，所以本章对文本数据进行了统计分析，以便更好地处理文本数据。本章还对数值数据进行了线性相关性分析，并基于 LightGBM 工具包提供的功能将重要性最低的两类特征进行了丢弃。最终，对文本数据进行清洗并分词之后，与数值数据一起按照一定的格式存储为训练数据集，便于后续章节的使用。

第 3 章 金融舆论文本词向量模型

由于文本数据无法直接被计算机处理，所以要采用某些方法将文本数据合理地表示为数字表示形式，以便计算机可以处理。本章主要简介了传统的词语表示方法，介绍了两种主流的词向量模型，并基于第 2 章所获得的文本数据采用两种不同的词向量模型进行了训练，从中选取了表现较好的模型。针对词向量模型在与情感相关的文本分析领域的缺点，本章基于前人的研究工作进行了改进，得到了一种针对本课题的词向量二阶段训练方法，实验结果表明该方法减轻了训练前的缺点。

3.1 传统的词表示法

3.1.1 词袋模型与 N 元模型

词袋模型[26]（Bag-of-words Model）是一种比较基础的文本数据表征方式，主要应用在自然语言处理以及信息检索领域。在词袋模型中，文本会被表示为词的聚合，将一组文本中的每一个均表示为等长向量，向量元素为在所有词中每个词出现的次数。为了简单起见，文本中的语法以及词的顺序均被抛弃。

下面是词袋模型表示方法的一个例子。如表中所示，这两段文本所组成的集合可以被称为语料库。词袋模型的第一个步骤是根据语料库建立词典，亦即对语料库中出现的每个词进行标号。若忽略标点符号，根据本语料库可以得到的词典为“今日”,“国内”,“市场”,“表现”,“普遍”,“不佳”,“美国”,“下跌”。根据得到的词典，可以对语料库中的文本进行表示，如表中所示。如第一个文本，因为“今日”出现过一次，所以向量的第一位取 1；“国内”出现过一次，所以向量第二位为 1，以此类推。

表 3.1 词袋模型的例子

Table 3.1 The examples of the bag-of-words model

原文（已分词）	词袋模型表示
今日 国内 市场 表现 普遍 不佳 。	[1,1,1,1,1,1,0,0]
美国 市场 普遍 下跌 。	[0,0,1,0,1,0,1,1]

在实际使用中，词袋模型主要用作是将文本数据转换为数值特征。将文本转化为词的集合之后，可以计算多种度量值作为文本表示特征向量的元素，比如最常见的就是上文使用的词频，或者采用词频-逆文档频率值（TF-IDF）。

词袋模型是一种无序的文档表示方法，其表征向量只表现了词的出现频数。例如在

上面的例子中，处理词袋模型表示时，模型无法获知动词“下跌”总是会出现在名词的后面。作为对词袋模型的补充，N 元模型（N-GRAM）可以存储空间信息，其具体方法为，采用一个长度为 N 的滑动窗口对原文本进行切分，在切分后的结果的基础上再采用词袋模型，得到向量表示。例如若采用二元模型，可以将语料库中的第一个文本切分为“今日国内”，“国内市场”，“市场表现”，“表现普遍”，“普遍不佳”，对语料库中所有文本均采取相同处理后，建立词袋模型。从概念上说，词袋模型可以被认为是 N 取 1 时的 N 元模型。

3.1.2 词语的向量表示

前文中提到的词袋模型与 N 元模型是将整个文本转化成一个特征向量，而如果需要采用细粒度的模型，即直接对词进行操作的话，需要一种词的表示方法。一种比较简单的表示方法被称为独热向量（One-hot Vector），具体过程为统计语料库中的所有文本并建立词典，将词表示为维度为词典容量的向量，并且向量中词对应的词典中的位置的位取值为 1，其他所有位置取值为 0。表 3.2 展示了独热向量的例子。首先仍然是对语料库构建词典，得到的词典为“国内”，“股市”，“上涨”，“美国”，“低迷”，然后原文可以表示为一组独热向量，独热向量维度为词典容量 5，如表中所示。

表 3.2 独热向量的例子

Table 3.2 The examples of one-hot vector

原文（已分词）	独热向量表示
国内 股市 上涨	[[1,0,0,0,0],[0,1,0,0,0],[0,0,1,0,0]]
美国 股市 低迷	[[0,0,0,1,0],[0,1,0,0,0],[0,0,0,0,1]]

独热向量表示法可以直接表示词语，并且表示简洁明了。但若采用独热向量表示法，容易造成数据有极大的稀疏性，并且如果语料库过大会造成词典容量过大，向量的维度会非常高，稀疏数据以及高维数据在深度学习模型中均难以处理；另外独热向量的词表示法没有捕捉到词语之间的相关性与相似性，例如在本例中，“上涨”与“低迷”两个词在句法中起到的作用是相似的，而独热表示法无法体现这一点。

3.2 词向量模型

针对词袋模型与独热表示的缺点，前人通过研究[27-28]提出了词嵌入模型（Word Embedding Model），也可称为词向量模型。“词嵌入”是指将一个词向量存在的高维离散空间嵌入到一个低维连续空间的过程，每个词均会被映射为实数域上的向量。高维离

散空间是指词语的独热表示形式，而低维连续空间是经过词嵌入处理后的词向量存在的空间。经过词嵌入处理后，得到的低维词向量并不再保持 0 或 1 的取值，而是实数域上的任意实数。词向量模型起源于 Bengio 等人[29]提出的神经概率语言模型，后人在此模型的基础上提出了 Word2Vec 模型[27-28]与 GloVe 模型[30]。

3.2.1 Word2Vec 模型

Word2Vec 模型是由 Mikolov 等人[27-28]研究的一种分布式词向量模型，其中主要包括两种框架，分别为连续性词袋模型（Continuous Bag-of-words Model，简称 CBOW）以及连续性跳跃元模型（Continuous Skip-gram Model，简称 Skip-gram）。作者在研究中提出，在向量空间中的词语的分布式表示有助于算法衡量不同词向量之间的距离，或对不同词语进行分组，从而在自然语言处理的任务中达到更好的性能。与之前所提出的神经概率语言模型不同，Word2Vec 模型避免了密集矩阵乘法，这使得训练过程非常高效。

CBOW 模型的结构图如图 3.1 左侧所示，其主要思想为，在一个句子中选取一个中心词，以中心词周围一定范围内的词作为特征，中心词类别作为模型预测目标。在训练完成之后可以将参数矩阵取出作为训练好的词向量表，整个模型为一个单隐层的前向神经网络，其中中心词左右的窗口范围为超参数。

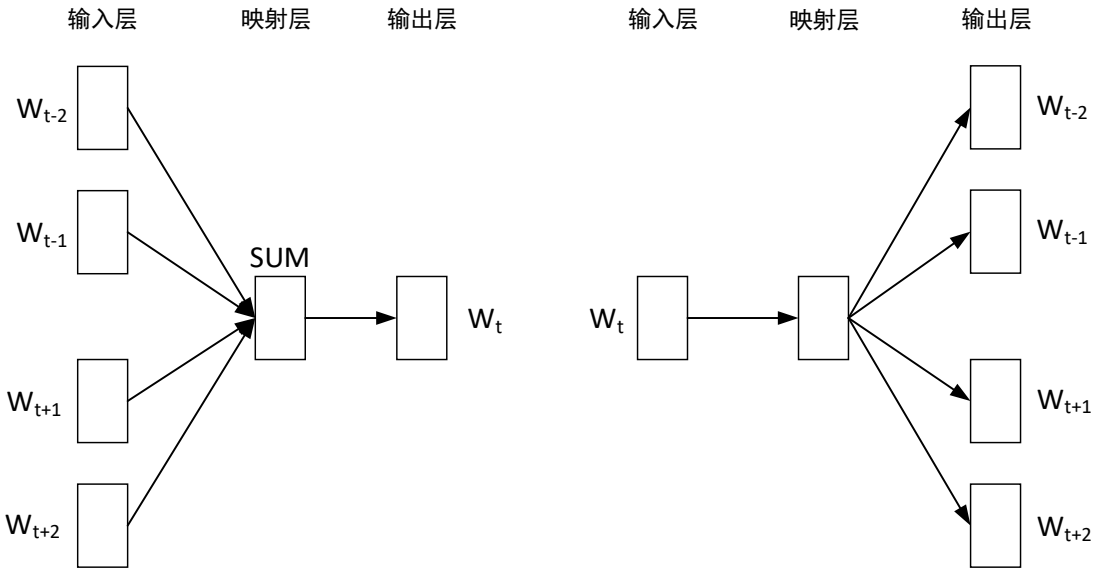

图 3.1 CBOW 模型（左）与 Skip-gram 模型（右）
Fig. 3.1 CBOW model (left) and Skip-gram model (right)

CBOW 模型的详细信息计算过程如图 3.2[31]所示，首先x_{1k}至x_{Ck}表示中心词的周围词的独热表示，其中词表维度为V，$W_{V\times N}$代表二维变量矩阵，训练完成后$W_{V\times N}$即为词向量表。x_{1k}至x_{Ck}分别与$W_{V\times N}$进行矩阵乘法，并按位求均值，得到隐层向量h，其维度为

N，是一个超参数。随后使用隐层向量与另一个变量矩阵$W'_{N\times V}$矩阵相乘后做 Softmax 处理得到输出层向量，取最大值的维度对应的词作为预测结果。

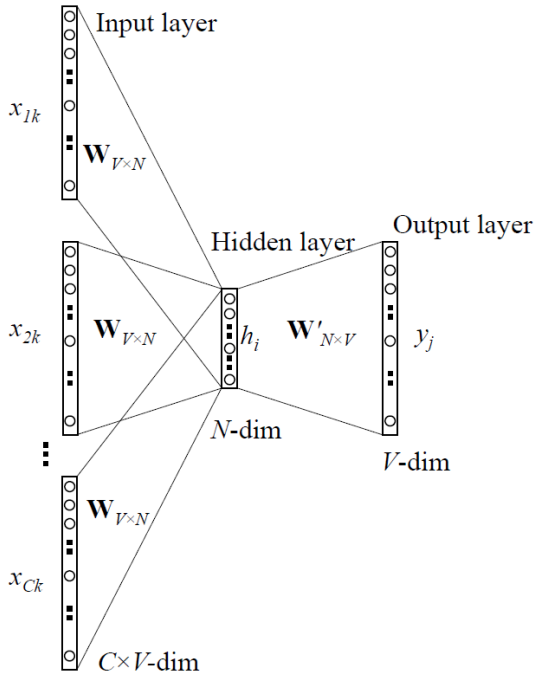

图 3.2 CBOW 模型计算过程

Fig. 3.2 The calculation process of CBOW model

CBOW 模型计算公式如下：

$$\mathrm{h} = \frac{1}{C}W^T(x_{1k} + x_{2k} + \cdots + x_{Ck})$$

$$= \frac{1}{C}(v_{1k} + v_{2k} + \cdots + v_{Ck})^T \tag{3.1}$$

$$\mathrm{y} = \arg\max_i \frac{e^{hW'}}{\sum_i e^{hW'}} \tag{3.2}$$

式中的加号代表按位相加，h代表隐藏层特征向量，C为窗口大小，W与W'均为权重矩阵，x_{1k}至x_{Ck}代表输入词的独热表示向量，v_{1k}至v_{Ck}为词的词向量表示，y为模型给出的中心词的预测类别。

CBOW 模型的损失函数如下:

$$\mathrm{L} = -\log p(w_O|w_{I,1},\cdots,w_{I,C}) \tag{3.3}$$

模型采用交叉熵损失函数，式中w_O为输出部分的词概率，$w_{I,1}$至$w_{I,C}$为输入部分的词概率,整个模型的训练采用梯度下降法减小损失函数值。

Skip-gram 模型结构图如 3.1 右侧所示，其主要思想与 CBOW 相反，将中心词作为特征预测周围的词的类别，训练过程完成后可以取出参数作为词向量表，整个模型仍然为一个单隐层前向神经网络。

Skip-gram 模型的详细计算过程如图 3.3[31]所示。x_k表述输入的中心词的独热表示向量，与上文类似，$W_{V\times N}$为二维变量矩阵，训练完成后$W_{V\times N}$即为词向量表。将x_k与$W_{V\times N}$进行矩阵相乘，得到隐层向量h，h再分别与$W'_{N\times V}$矩阵相乘后做 Softmax 处理，得到y_{1j}至y_{Cj}。y_{1j}至y_{Cj}表示模型对中心词周围词的预测类别。

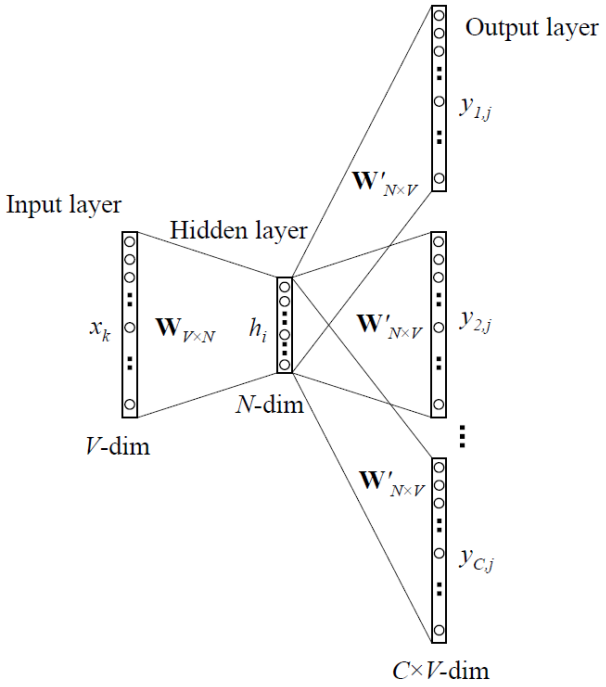

图 3.3 Skip-gram 模型计算过程

Fig. 3.3 The calculation process of Skip-gram model

Skip-gram 模型计算公式如下：

$$\mathrm{h} = x_k W \tag{3.4}$$

$$\mathrm{y}_c = \arg\max_i \frac{e^{hW'}}{\sum_i e^{hW'}} \tag{3.5}$$

式中的h代表隐藏层特征向量，C为输出数据窗口大小，W与W'均为权重矩阵，x_k代表输入词的独热表示向量，y_c为模型给出的周围词的预测类别，所以每次计算时会生成C个输出。

与 CBOW 模型相似，Skip-gram 模型也采用交叉熵损失函数，模型的损失函数如下：

$$\mathrm{L} = -\log p(w_{O,1},\cdots,w_{O,C}|w_I) \tag{3.6}$$

式中w_I为输入部分的词概率，$w_{O,1}$至$w_{O,C}$为输出部分的词概率，整个模型的训练采用梯度下降法减小损失函数值。

由于模型结构存在较大差异，所以在训练过程中 CBOW 速度较快而 Skip-gram 较慢，但 Skip-gram 在低频词上的表现更好。由 Word2Vec 训练得到的词向量有一些特性，

这些特性说明学习到的词向量表示可以学习到某些自然语言领域的规则与模式，如某些词向量相加后会产生有意义的结果，以及某些类别中的词与另一类别中词之间有相似关系。

3.2.2 全局向量词表示模型

全局向量词表示模型（Global Vectors for Word Representation，简称 GloVe），由 Pennington 等人[30]研究并发表，其与 Word2Vec 最大的不同之处在于其摒弃了深度学习模型，直接采用梯度下降法学习词向量。具体过程分为两步，首先对语料库构建共现矩阵，假设词表容量为 V，则该共现矩阵为一个维度为 V 的方阵，其中的元素为两个词在语料库中出现在同一窗口内的次数，窗口大小为超参数。接下来采用梯度下降法降低损失函数，损失函数的定义式如下：

$$\mathrm{L} = \sum_{m,n}^{N} f(X_{m,n})(v_m^T v_n + b_m + b_n - \log(X_{m,n}))^2 \tag{3.7}$$

其中N为词典容量大小，v_m与v_n是词语m与词语n的词向量，b_m与b_n为标量，代表词向量对应的偏差项，$X_{m.n}$为共现矩阵中的对应元素，代表两个词语同时出现在一个窗口中的次数。$f(X_{m.n})$为权重函数，词频越高函数值越大，如式（3.8）所示，式中$xmax$为分割阈值。

$$f(x) = \begin{cases} (\frac{x}{xmax})^{0.75}, & if\ \ x < xmax \\ 1, & if\ \ x \geq xmax \end{cases} \tag{3.8}$$

Word2Vec 本质上采用的是局部上下文窗口，学习的仅是区域特征，缺乏整体的词语之间的关系；而 GloVe 融合了全局统计信息与局部上下文窗口，与 Word2Vec 差异较大；另外，GloVe 模型在训练过程中还抛弃了神经网络模型，这使得其训练速度可以大幅提升。

3.3 词向量训练加速

3.3.1 Word2Vec 模型训练的问题

Word2Vec 模型虽然在结构上简化了神经概率语言模型，但仍然没有解决词向量训练缓慢的根源问题，即输出层维度较高。此处以 CBOW 模型举例，假设语料库词典容量为十万，则输入层与输出层的维度均为十万。由于训练目的是希望获取代表词向量表的二维变量矩阵，所以无法对输入层进行化简；在输出层，由于 Softmax 与交叉熵损失函数的特性，导致梯度反向传播过程的计算量非常大，进而造成训练过程缓慢。为了解

决上述问题，可以采用分层 Softmax（Hierarchical Softmax）方法与样本负采样（Negative Sampling）方法，本节对这两种方法进行介绍，并将其应用于词向量训练过程中。

3.3.2 分层 Softmax

分层 Softmax 对输出层进行优化，原始模型在输出层采用 Softmax 计算概率值，而分层 Softmax 利用霍夫曼树计算概率值。具体来说，将词表中的每一个词都作为霍夫曼树中的叶子结点，以词在语料库中的词频作为节点权重构建霍夫曼树作为输出。霍夫曼树是一个二叉树，在叶子节点与其权重确定时，树的权重和路径长度最短。也就是叶子节点的权重越高该节点距离根节点就越近，权重较高的点代表了词频较高的词。在一个霍夫曼树中，从根节点出发，到达某一叶节点的路径具有唯一性，分层 Softmax 正是利用了这一特性来简化计算。

图 3.4[31]描述了一个根据给定语料库词典建立的霍夫曼树，其中白色节点w_1至w_V为叶节点，代表各个词，灰色节点为分支节点，其权值为此分支节点的所有叶节点的权重和。此处以w_2为例，从根节点到w_2的路径长度为$\mathrm{L}(w_2)=4$，路径为黑色标记的线段。图中的非叶节点也会学习其相应的向量表示，原因在于分层 Softmax 将原本的 V 类分类问题转化为多个二分类问题，每个二分类问题对应于霍夫曼树中的一个分支点，在训练过程中，若二分类完成命中则停止分类过程。

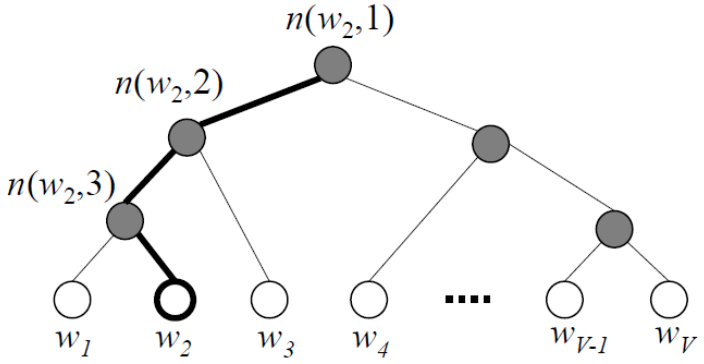

图 3.4 分层 Softmax 中的霍夫曼树

Fig. 3.4 Huffman tree in hierarchical softmax

此处以 CBOW 模型举例，输出层不断采用二项逻辑斯蒂回归进行二分类，则在霍夫曼树中搜索到某一个分叉点时，选择左子节点与右子节点的概率分别为：

$$P_l = \sigma(u_{n(w,j)}^T \hat{v}_t) \tag{3.9}$$

$$P_r = \sigma(-u_{n(w,j)}^T \hat{v}_t) \tag{3.10}$$

式中$\hat{v}_t$为输出层输入，$u_{n(w.j)}$代表此时分离点向量表示，σ代表 Sigmoid 函数。

霍夫曼树中的词的概率可以表示为一个条件概率，其代表着由一条从根节点到这个词对应的叶节点的路径，将式（3.9）与（3.10）统一，则树中词的条件概率可以表示为：

$$\begin{aligned}&\mathrm{P}(\mathrm{w}|w_{t-m},\cdots,w_{t-1},w_{t+1},\cdots,w_{t+m})\\&=\prod_{j=1}^{L(w)-1}\sigma([\![n(w,j+1)=ch(n(w,j))]\!]u_{n(w,j)}^{T}\hat{v}_t)\end{aligned}\tag{3.11}$$

其中$L(w)-1$代表分离点之前经过的节点数，$[\![\cdot]\!]$代表内部表达式为真时取 1，为假时取-1。对于 Skip-gram 模型，同样可以得到：

$$\begin{aligned}&\mathrm{P}(\mathrm{w}|w_{t-m},\cdots,w_{t-1},w_{t+1},\cdots,w_{t+m})\\&=\prod_{i=1}^{2m}\prod_{j=1}^{L(w)-1}\sigma([\![n(w,j+1)=ch(n(w,j))]\!]u_{n(w,j)}^{T}\hat{v}_{t+i})\end{aligned}\tag{3.12}$$

分层 Softmax 平均时间复杂度为$\mathrm{O}(\log(|\mathrm{V}|))$，相比单纯使用 Softmax 的时间复杂度$|\mathrm{V}|$有很大提升。

3.3.3 样本负采样

采用分层 Softmax 可以提高模型的训练效率，但如果是在处理生僻词时，模型需要在霍夫曼树中搜索相当长的时间。样本负采样方法摒弃了霍夫曼树，其主要思想为，将语料库中的语料的一个子字符串的中心词替换为其他词，虚构出一个语料库中不存在的语料。在这种处理下，优化目标变为了最大化正样本概率并且减小虚构样本的概率。根据上文所描述的方法，采用逻辑斯蒂回归模型对某个样本是正样本的概率进行建模：

$$\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})=\sigma(u_w^T\mathrm{v}_c)\tag{3.13}$$

其中w代表中心词，c代表中心词周围的上下文词，u_w与v_c分别代表两者对应的词向量。所以对于全部正样本与负样本，似然函数分别为：

$$\prod_{(w,c)\in POS}\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})\tag{3.14}$$

$$\prod_{(w,c)\in NEG}\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})\tag{3.15}$$

其中POS与NEG分别代表正样本集合与负样本集合。若要使式（3.14）增大并减小式（3.15），等价于对如下似然函数最大化：

$$Likehood=\prod_{(w,c)\in POS}\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})\prod_{(w,c)\in NEG}(1-\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c}))\tag{3.16}$$

所以最终的对数似然函数为：

$$\begin{aligned}&\mathrm{L}=\log(\prod_{(w,c)\in POS}\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})\ \prod_{(w,c)\in NEG}(1-\mathrm{P}(\mathrm{D}=1|\mathrm{w},\mathrm{c})))\\&=\sum_{(w,c)\in POS}log\,\sigma(u_w^T\mathrm{v}_c)+\sum_{(w,c)\in NEG}log\,\sigma(-u_w^T\mathrm{v}_c)\end{aligned}\tag{3.17}$$

对似然函数取反就可以得到模型的损失函数，之后采用梯度下降法进行学习。

3.3.4 高频词降采样

在某些较大的语料库中，一些高频词可能会出现上千万次，比如“的”，“了”等词。通常情况下这一类词所提供的信息有限，比如相比于“股票”，“的”两个词的共现，模型能从“股票”与“上涨”之间的共现学习到更多的信息。另外，在训练过程中，高频词在经过一段时间的训练之后，其词向量表示就已经不再发生改变。

为了解决生僻词与高频词之间的不平衡，需要在训练词向量过程中对高频次进行降采样，具体方法为对于每一个词，按照如下公式计算出的概率进行随机舍弃：

$$\mathrm{P}(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{3.18}$$

式中$f(w_i)$为单词w_i的频率，t是选择的阈值，通常取10^{-5}左右。降采样方法提高了学习速度，并且显著提高了罕见词的向量表示的准确性。

3.4 基于金融文本语料库的词向量训练

3.4.1 实验环境与参数选择

本文采用 Gensim 与 GloVe 工具包训练了两种不同的词向量，分别对应上文所述的 Word2Vec 模型与 GloVe 模型，并在本节中比较两者的表现。Gensim 是一个自然语言处理工具包，其中包含了多种功能，包括基于 Word2Vec 模型的词向量训练功能，该工具包计算速度快，使用简单；GloVe 工具包由 GloVe 模型的提出者的团队开发，主要功能为实现了基于 GloVe 模型的词向量训练方法。本节的实验环境如表 3.3 所示。

表 3.3 词向量训练实验环境

Table 3.3 Experimental environment of training word vectors

操作系统	编程环境	词向量工具包	CPU
Windows 10	Python 3.6	Gensim 0.4.0, GloVe 1.0.0	i7-7700HQ

词向量训练采用的语料库为经过文本数据清洗、特定字符替换与文本分词处理之后得到的所有文本组成的集合，处理过程如第 2 章所述，语料库中共计包含 7102316 条文本数据，采用 Gensim 的 Skip-gram 模型训练 3.3 小时，GloVe 训练 2.4 小时，训练过程采用的主要参数设定如下表 3.4 所示。其中加速方法为上一节中的针对 Word2Vec 模型的改进方法，所以表中 GloVe 模型加速方法一项为空。Gensim 中的窗口宽度是指 Skip-gram 中采用的上下文窗口，代表在中心词周围各取 2 个值；GloVe 模型中的窗口宽度为共现窗口，用于统计共现情况，所以窗口宽度较大。

表 3.4 主要参数

Table 3.4 The main parameters

工具包	词向量维度	初始学习率	学习轮数	窗口宽度	加速方法
Gensim	300	0.025	5	5	样本负采样，高频词降采样
GloVe	300	0.05	5	10	-

3.4.2 实验结果分析与比较

在经过上述训练过程之后，本文得到了两个不同的词向量模型。本小节对两个词向量模型进行对比评估，并选择其中表现较好的模型作为后续预测模型的一部分。由于词向量模型为广义上的无监督学习模型，亦即所采用的训练数据没有标签，由于这种特殊情况，导致词向量模型没有定量的评估标准，所以本文采取对几个典型词求整个词典中的最相似词列表，并采用人工标准对相似列表中的词与目标词的词义进行评价。本小节以三个词为例，分析两种模型的表现差异，选取的三个词为“上涨”，“持平”，“市场”。

图 3.5 采用 Python 集成开发环境 Spyder 的数据浏览功能，展示了 Word2Vec 模型与 GloVe 模型中与“上涨”最相近的 10 个词，括号中第二位为本词与目标词的余弦相似度，图中按照余弦相似度降序对这 10 个词进行了排序。由图中可以看到两者列表所含词的词义差距不大，并且在两个列表中均出现了反相情感词，如“下跌”。

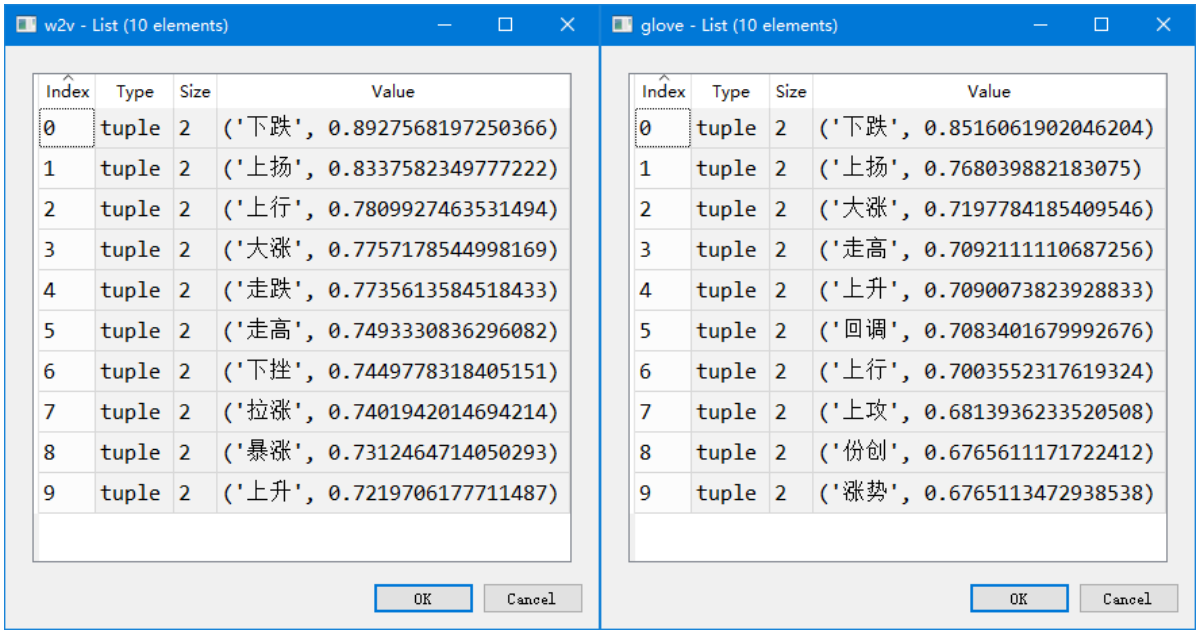

图 3.5 Word2Vec 模型（左）与 GloVe 模型（右）中与“上涨”最相似的词

Fig. 3.5 The most similar words of "上涨" in the Word2Vec model (left) and GloVe model (right)

图 3.6 展示了两模型中与“持平”最相近的 10 个词。由图中可以看到左侧的

Word2Vec 模型中的最相似词列表与“持平”语义相似度较高，而在 GolVe 模型中出现了一些在语义上与“持平”相关性不大的词以及无意义的不合理切分词，比如“干吨”，“焦炼”等词。

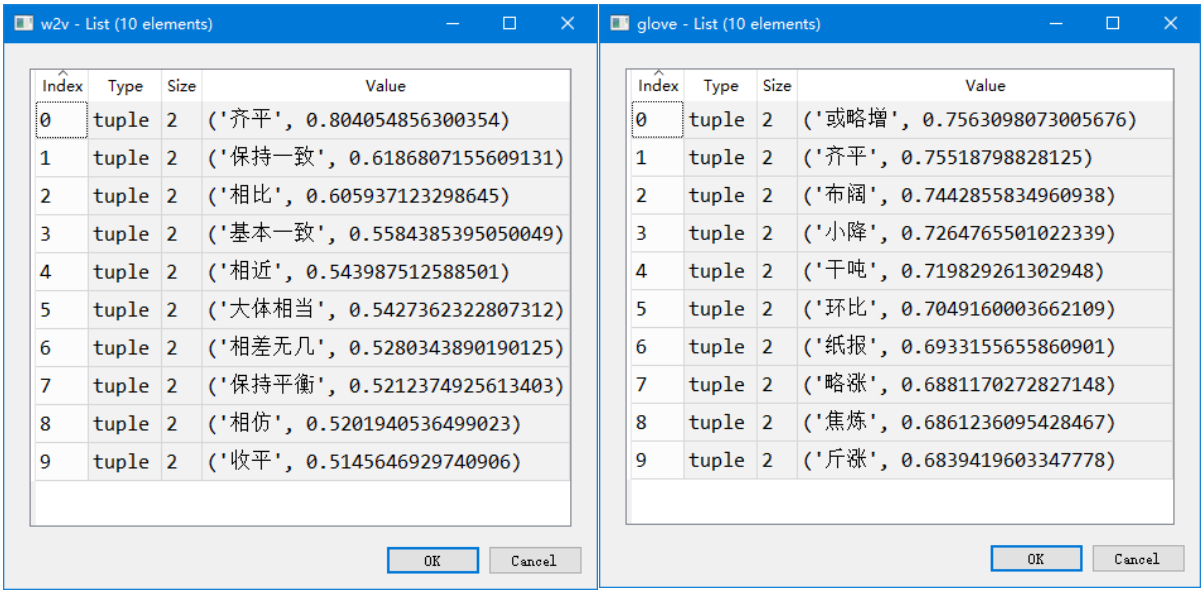

图 3.6 Word2Vec 模型（左）与 GloVe 模型（右）中与“持平”最相似的词

Fig. 3.6 The most similar words of "持平" in the Word2Vec model (left) and GloVe model (right)

图 3.7 展示了两模型中与“市场”最相近的 10 个词。由图中可以看到左侧的 Word2Vec 模型中的最相似词在词义上均与“市场”较为相似，而 GloVe 模型中仍然出现了一些不相关词与不合理切分词。

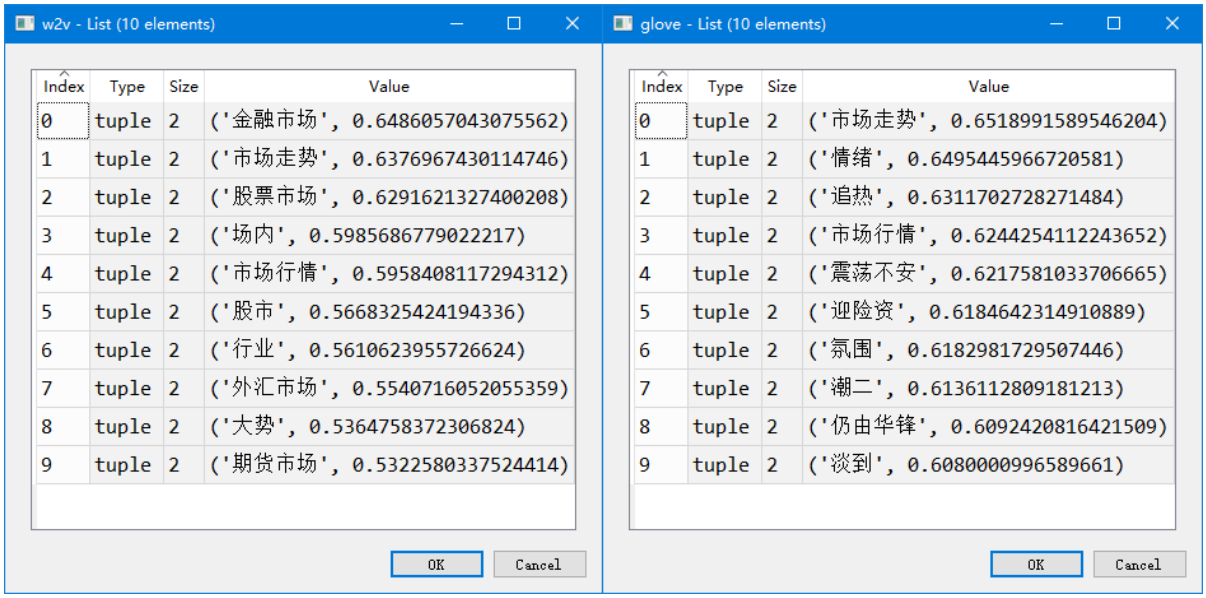

图 3.7 Word2Vec 模型（左）与 GloVe 模型（右）中与“市场”最相似的词

Fig. 3.7 The most similar words of "市场" in the Word2Vec model (left) and GloVe model (right)

经过在三个词的基础上对两模型的比较，可以发现 Word2Vec 模型的表现明显优于

GloVe 模型，所以本文选取由 Word2Vec 模型训练得到的词向量作为后续模型的输入。

3.5 基于金融词典的词向量调整

3.5.1 词向量模型的问题

在通常情况下，词向量模型可以高效地捕捉词语蕴含的语义信息以及语法信息，并且在某些自然语言处理领域如机器翻译与文本分类领域表现极佳；但词向量模型在情感分析领域有一个严重的缺陷，就是其无法捕捉到文本的情感信息[32]。如在上一节中，与“上涨”最相似的词在两个模型中均为“下跌”，这种结果会使得进行情感分析的模型对词语的情感倾向产生混淆。图 3.8 展示了 Word2Vec 模型中与“上涨”及“下跌”两个词的最相似词列表，在左图中，“下跌”、“走跌”、“下挫”被模型错误地认为与“上涨”相似度较高；在右图中，“上涨”、“回调”、“上扬”被模型错误地认为与“下跌”相似度较高。

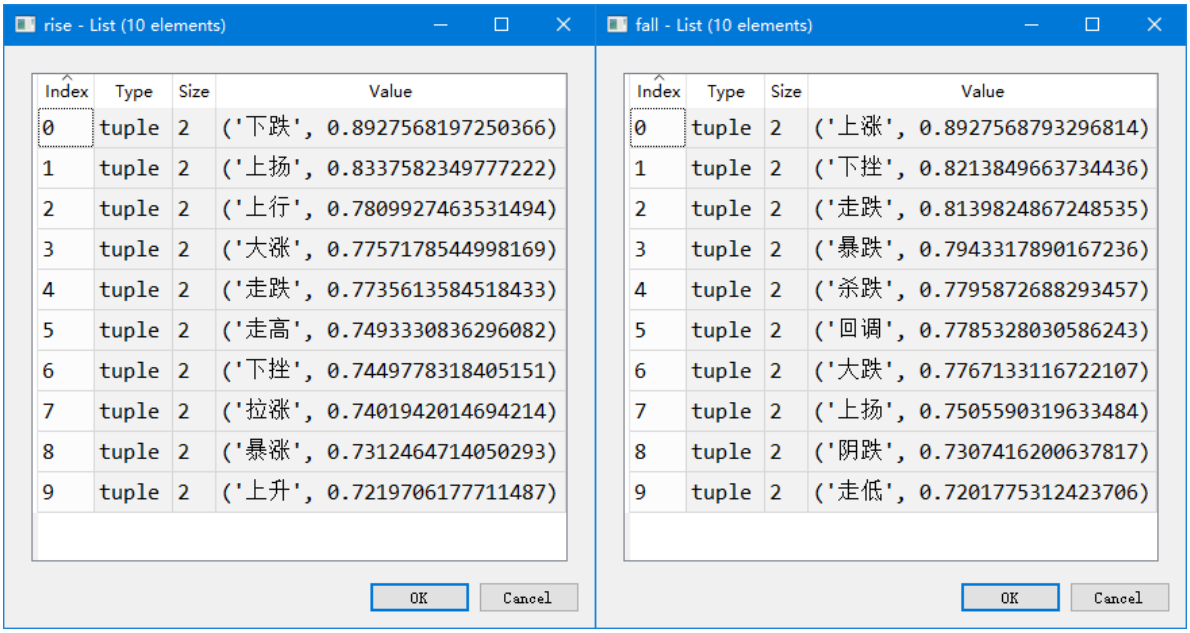

图 3.8 “上涨”（左）与“下跌”（右）的最相似词列表

Fig. 3.8 The list of most similar words to "上涨" (left) and "下跌" (right)

为了缓解这种情况，使词向量对词语情感的表示更为精确，需要在已训练得到的词向量模型的基础上进行词向量二阶段训练。

3.4.2 词向量二阶段训练的方法

词向量二阶段训练是指，在第一阶段训练得到的词向量的基础上，针对具体的任务

目标再次训练词向量，通常需要二阶段训练的场景主要是情感分类。经过二阶段训练过程得到的词向量通常在应用场景上有更好的表现。词向量二阶段训练方法主要有两类，第一类是构建一个新的有监督学习模型，此时需要数据有标签，该模型与上文介绍的 CBOW 模型较为相似；第二类是基于某个目标函数直接采用优化方法对词向量进行调整，通常此类模型需要外置词典的帮助。下面对两类方法进行介绍。

图 3.9 展示了第一种方法的流程，假设应用背景为一个二分类的情感分类任务，其中w_1至w_n为语料中的各个词的独热向量表示，W为一个二维矩阵，与 CBOW 中类似，此处对应着词向量表，其中V为词典容量，N为词向量维度，但与 CBOW 中不同的是，此处的W不再采取随机数进行初始化，而是采用第一次训练得到的词向量表进行初始化。另外，在这个训练任务中，模型的目的是预测语料的情感类别，类别数为 2，而在初次训练词向量的过程中，类别数为V。

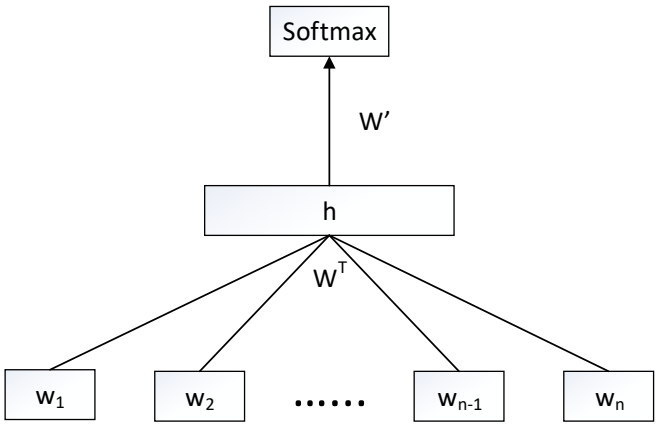

图 3.9 基于监督学习的词向量二阶段训练

Fig. 3.9 Second-stage training of word vectors based on supervised learning

该模型的计算过程如下：

$$\mathrm{h} = \frac{1}{n}\sum_{i=1}^{n}(w_1, \cdots, w_n)W^T \tag{3.19}$$

$$\hat{y} = argmax(Softmax(hW')) \tag{3.20}$$

其中W与W'均为权重矩阵，并且W采用预训练词向量表进行初始化，w_1至w_n为各个词的独热向量表示，h为隐藏层向量，由各个词向量按位平均获得，最终模型输出预测的情感类别$\hat{y}$。模型采用交叉熵损失函数，如下所示：

$$L = y log\hat{y} + (1-y)\log(1-\hat{y}) \tag{3.21}$$

虽然第一种方法简洁明了且易于操作，但由于本文所收集到的文本数据不含标签，所以无法采用第一种方法进行训练。而第二种方法直接对原词向量进行调整，无需采用监督学习过程，也就不需要标记数据，但需要一个外置辅助词典。本方法由 Yu 等人[33]研究并发表，其主要思想是，以一个以不同情感类别分类的词典，挑选出需要

调整的一些词，若某个词在词嵌入空间中与反向倾向类别词距离较近而与同向倾向类别词距离较远，则采用迭代算法调整这个词的位置，使其向同向倾向类别词靠拢。

Yu 等人在文中基于一个情感二分类任务，采用一个包含极性分数的情感词典对词向量进行调整。图 3.10[33]展示了一个例子，即对词“good”进行调整，图中括号内的数值代表这个词的情感极性，数值介于 0 到 10 之间，数值越大词越倾向于正向极性，数值越小词越倾向于负向极性。整个算法分为两个部分，第一步称为重排序，首先针对目标词，在第一次训练得到的词向量中选取与目标词距离最近的 k 个词，本例中 k 取 10，测量标准为余弦相似度；其次对所得到的词表重新排序，首先判断目标词的类别，若目标词为正向词，即其情感极性大于 5，则重新排序时依照情感极性降序排序；若目标词情感极性小于 5，则依照情感极性升序重新排序。

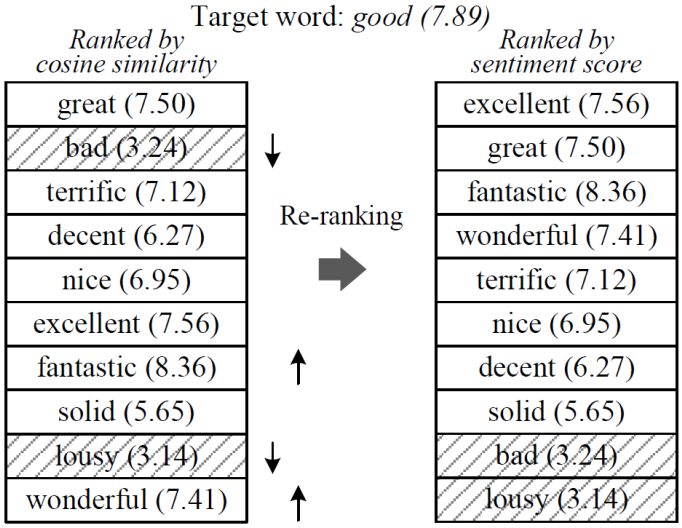

图 3.10 重排序过程

Fig. 3.10 Resequencing process

在经过重新排序后，模型会构造一个最优化问题并求其次优解。优化问题的目标函数如下：

$$\Phi(V) = \sum_{m=1}^{l} \sum_{n=1}^{k} w_{mn} dist(v_m, v_{mn}) \tag{3.22}$$

其中外循环代表对目标词库进行的循环，l为目标词数量，内循环为目标词的近邻循环，k代表模型中所设定的近邻数，v_m为目标词，v_{mn}为目标词的近邻词，$\mathrm{dict}(v_m, v_{mn})$为目标词与近邻词的距离，$w_{mn}$为近邻词的权重，取值为其在词表中的排序值的倒数。由于优化方法采用梯度下降法，所以为了计算简便，算法采用了平方欧几里得距离，公式如下，其中D代表词向量维度：

$$\mathrm{dist}(v_m, v_{mn}) = \sum_{d=1}^{D} (v_m^d - v_{mn}^d)^2 \tag{3.23}$$

词向量调整的示意图如图 3.11[33]所示。

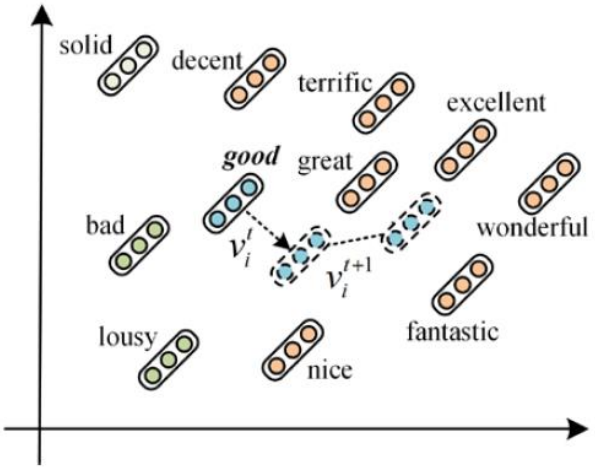

图 3.11 词向量调整的过程（以“good”为例）

Fig. 3.11 The process of word vectors adjustment (take "good" for example)

若调整后的词向量与调整前的词向量的距离较大，则可能会对词向量的语义发生较大改变；为了防止这种情况的发生，算法的目标函数中加入惩罚项，惩罚项的目的是使每一次迭代过程中词向量的变化不太大。加入惩罚项后的目标函数表达式如下：

$$\Phi(V) = \sum_{i=1}^{n}[\alpha dist(v_i^{t+1}, v_i^t) + \beta \sum_{j=1}^{k} w_{ij} dist(v_i^{t+1}, v_{ij}^t)] \tag{3.24}$$

式中第一项为惩罚项，其中α与β为超参数，用来调节惩罚项与原目标函数的占比。

3.4.3 基于金融词典的词向量二阶段训练

上文介绍了两种词向量二阶段训练的方法。由于本文中所采集到的文本数据虽然有情感倾向，但不属于有标签数据，所以只能采用第二种方法进行二阶段训练。针对第二种方法所需的外部词典，文本收集了约 132 个与股票涨跌走势最相关的倾向词，分为乐观倾向 53 词，中立倾向 34 词与悲观倾向 45 词，部分词语如表 3.5 所示。本部分对第二种词向量调整方法进行修改，使其更适合本文情景，并使用本方法对目标词表中的词进行二阶段训练。

表 3.5 部分目标词举例

Table 3.5 Some examples of target words

倾向	部分目标词
乐观倾向	上涨，暴涨，涨停，上扬，反弹，强劲，回升，利好
中立倾向	震荡，缓慢，萎靡，前景不明，低迷，调整
悲观倾向	下跌，暴跌，回落，回调，跌停，退却，下滑

由于 Yu 等人在研究中采用的是带有情感倾向值的英文词典，而在中文领域暂无此类数据，所以文本对原方法中的近邻重排序过程做了部分修改，新方法为：若目标词为

乐观倾向，则按照“乐观|中立|悲观”重新排序；若目标词为中立倾向，则按照“中立|悲观|乐观”重新排序；若目标词为悲观倾向，则按照“悲观|中立|乐观”重新排序。排序后子类中的词序保持原顺序不变。对于超参数k，即近邻的个数，本文取 10，近邻的选取可以从所有词中选择，不一定是在大小为 132 的目标词集中。对于近邻词的权重w_{mn}，按照排序类别，第一类与第二类取$\frac{1}{m}$，m为其在整个序列中的排序，第三类取$-\frac{1}{m}$。对比原方法，此处改进了异类词权重值，使目标词向其反方向运行。对“上涨”的 10 个最近邻重排序的过程以及各项权重如图 3.12 所示。其中橙色代表乐观倾向词，绿色代表悲观倾向词，重排序后各子类内顺序不变。

原排序
下跌
上扬
上行
大涨
走跌
走高
下挫
拉涨
暴涨
上升

重排序	权重
上扬	1
上行	1/2
大涨	1/3
走高	1/4
拉涨	1/5
暴涨	1/6
上升	1/7
下跌	-1/8
走跌	-1/9
下挫	-1/10

图 3.12 对“上涨”近邻词的重排序

Fig. 3.12 Reordering for nearest neighbor words of "上涨"

由于对异类词的权重进行了修改，导致模型在梯度下降中的梯度值可能过大，为了解决这个问题，本文在改进中采用了梯度截断。具体来说就是将所求得的梯度除以梯度的模。修改前与修改后的迭代更新公式如下：

$$v_{t+1} = v_t - a\nabla f(v_t) \tag{3.25}$$

$$v_{t+1} = v_t - a\frac{\nabla f(v_t)}{||\nabla f(v_t)||} \tag{3.26}$$

其中v_{t+1}代表迭代至$t+1$次的词向量，v_t为t次迭代时的词向量，a为迭代的步长，$\nabla f(v_t)$为目标函数在v_t处的梯度。

图 3.13 展示了词向量初次训练的模型中“上涨”与前 10 个最近邻词的空间相对关系，图 3.14 展示了词向量二阶段训练之后的空间相对关系，图中显示的点是将 300 维的词向量使用 PCA 降维至 2 维空间的结果，可以看到，相比于初次训练的结果，二阶

段训练后目标词更接近同类别词。

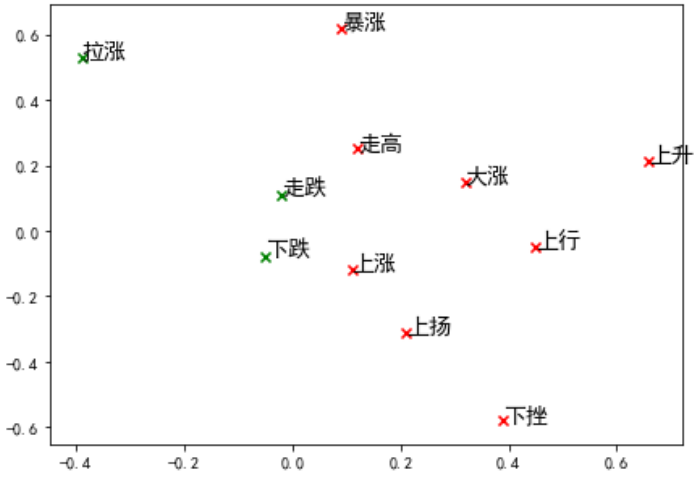

图 3.13 初次训练后目标词与 10 个最近邻词的位置

Fig. 3.13 The position of the target word and the 10 nearest neighbors after the initial training

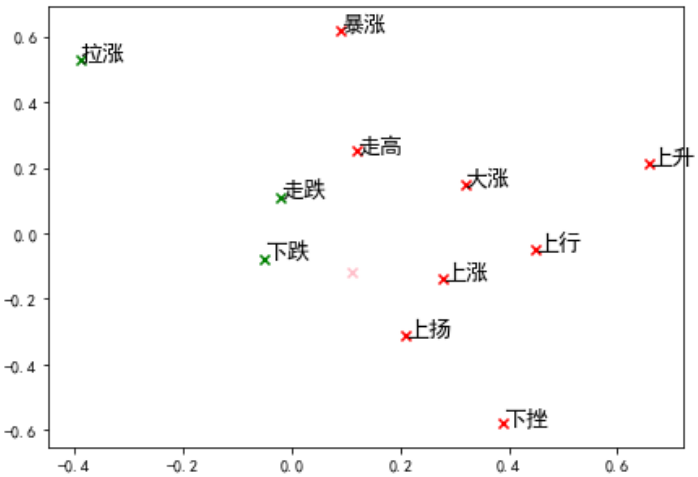

图 3.14 二阶段训练后目标词与 10 个最近邻词的位置

Fig. 3.14 The position of the target word and the 10 nearest neighbors after the second-stage training

3.6 本章小结

本章主要介绍了本文中词向量模型相关的内容。首先介绍了词向量模型的概念与两种主流的词向量模型，以及词向量模型的加速训练方法；之后基于已获取的文本数据训练了两个不同的词向量模型，并采用词义相似性选择了表现较好的 Word2Vec 模型训练出的词向量作为预测模型的输入。

由于词向量模型存在着无法准确表征带有情感倾向的问题，这会对预测模型带来不利影响；本文基于前人提出的针对文本情感分类场景的词向量二阶段训练方法，针对本文中的应用背景做了部分必要的改进，实验表明经过二阶段训练后的词向量对情感倾向的表征能力有所改善。

第 4 章 基于深度学习的金融文本模型建立

由于文本的研究中包含大量的文本数据，传统的机器学习学习模型难以对其进行建模；而近年来深度学习技术发展迅速，在图像、文本、音频视频等应用领域均有突破，这为本文处理文本数据提供了思路。本章将介绍预测模型中所采用的深度学习模块，并基于分层结构设计了多个模型，之后通过实验对比，在兼顾训练时间与精度的情况下选择了其中的一个文本模型，该模型为最终预测模型的一部分。

4.1 神经网络

4.1.1 神经元与激活函数

神经元是神经网络[34-35]的基本组成部分，其主要作用是对输入数据进行加权求和，进行非线性变换后将其输出。神经元的示意图如图 4.1 所示，其输入数据为x_1, x_2, x_3，$+1$为偏置项。

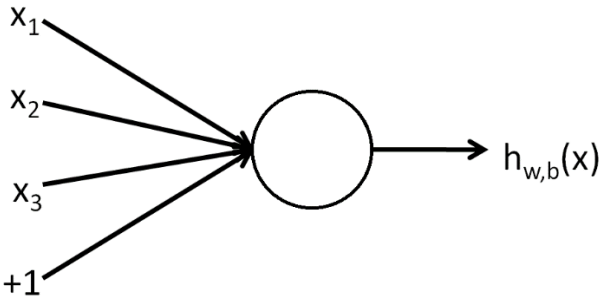

图 4.1 神经元结构图

Fig. 4.1 Neuronal architecture

神经元的计算公式如下：

$$h_{w,b}(x) = f(W^T x) = f(\sum_{i=1}^{3} W_i x_i + b) \tag{4.1}$$

式中W^T代表各输入项对应的权重，此类值在模型训练时先采用某种策略进行初始化，再通过梯度反向传播方法进行调整，直至达到较好的输出结果。式中b为偏置项，对比于图中的$+1$；$f(\cdot)$为神经元的激活函数，其功能是对数值进行非线性映射。

目前主流的激活函数有三种，分别为 Sigmiod 函数，tanh 函数与线性整流函数[36-37]（ReLU），其表达式如式（4.2）到（4.4）所示，函数曲线分别如图 4.2 所示。

$$f(z) = \frac{1}{1+e^{-z}} \tag{4.2}$$

$$f(z) = \frac{1-e^{-z}}{1+e^{-z}} \tag{4.3}$$

$$f(z)=\begin{cases} z, & z>0 \\ 0, & z\le 0 \end{cases} \tag{4.4}$$

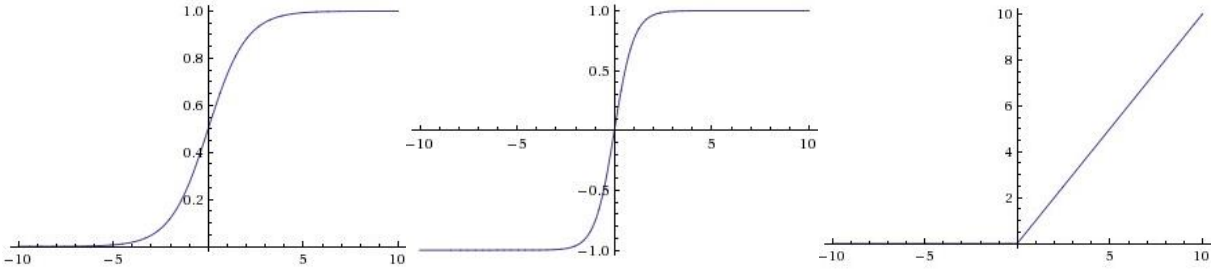

图 4.2 Sigmoid 函数（左），tanh 函数（中），ReLU 函数（右）

Fig. 4.2 Sigmoid function (left), tanh function (middle), ReLU function (right)

由图可以看出，Sigmoid 函数与 tanh 函数在较小的定义域内波动幅度较大，亦即函数的一阶导数值较大，而在大部分定义域内函数值波动较小，亦即一阶导数值较小，这种激活函数的性质容易导致神经网络产生梯度消失与等问题，也就是在经过一段时间的训练之后，由于损失函数的梯度消失导致无法获得下降梯度，模型无法进行学习过程。ReLU 激活函数的提出解决了这一问题，其由两个函数拼接而成，$(0,0)$点处的导数未定义。由于其定义域内不存在导数饱和的区域，所以降低了神经网络中梯度消失的风险；另外，其表达式较为简单，使得神经网络的计算过程以及梯度反向传播过程的速度大大提升。

由于 ReLU 中自变量小于 0 时的函数部分为常函数，导数为 0，所以当神经元输入值加权求和后结果小于 0 时，神经元的权值不再被更新，这种状态被称为神经元“死亡”状态。由于 ReLU 激活函数的缺点，研究者们提出了多种改进版本的 ReLU 激活函数[38]，主要工作为对其常函数部分进行优化，并且将$(0,0)$点处变更为可导点。虽然有上述两个缺陷，但 ReLU 激活函数仍然为最常用的激活函数之一。

4.1.2 全连接神经网络

全连接神经网络[34-35]（Fully Connected Network，简称 FCN）是最基本的神经网络模型，其他高级结构均是由 FCN 改进与变形而来。将上文介绍的神经元横向排列组成一个向量，就是神经网络的一层，由多个神经网路层堆叠之后，并对网络输入部分与网络输出部分进行特殊处理，就可以得到一个全连接神经网络。图 4.3 展示了一个三层神经网络，图中的圆圈代表神经元，图中的第一层为网络的输入层，包含三个单元，负责接收输入数据，即图中的x_1, x_2, x_3，$+1$为偏置项；第二层代表网络的隐藏层，包含三个单元，负责对输入层的数据进行加权求和后做非线性变换，隐藏层中的神经元功能与上文介绍的相同；第三层代表网络的输出层，由于本例是一个回归任务，所以输出层只有

一个神经元。

一个全连接神经网络只有一个输入层与一个输出层，而隐藏层可以有多个。隐藏层的命名来源是由于神经网络中非输入输出层的值无法被观测。由图中可以看出，神经网络的某层的每一个单元与前一层的所有单元之间均有连接，所以被称之为全连接神经网络。

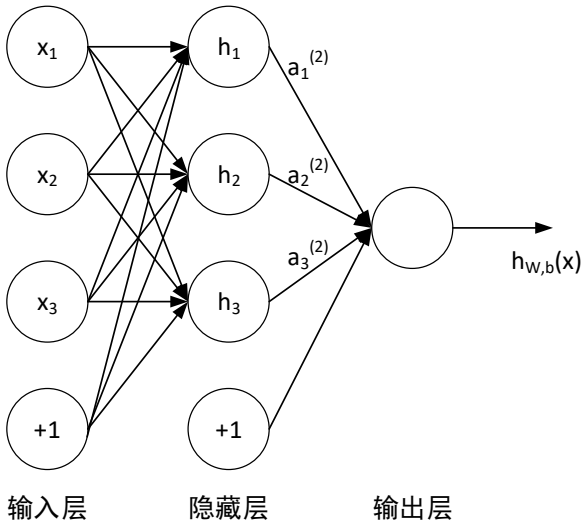

图 4.3 全连接神经网络

Fig. 4.3 Fully connected neural network

图中网络的计算过程可以表示为：

$$a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \tag{4.5}$$

$$a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \tag{4.6}$$

$$a_3^{(2)} = f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)}) \tag{4.7}$$

$$h_{W,b}(x) = a_1^{(3)} = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)}) \tag{4.8}$$

式中$a_m^{(l)}, b_n^{(l)}$分别表示第l层的第m个单元的输出值与偏置值，$W_{mn}^{(l)}$表示m层与n层之间的权重矩阵，$h_{W.b}(x)$为模型的输出值。

4.2 文本卷积神经网络

4.2.1 卷积神经网络

卷积神经网络（Convolutional Neural Network，简称 CNN），最早由 LeCun 等人[39]提出，是一种专门用来处理具有平行特征数据的神经网络结构，如时间序列数据与图像数据。卷积神经网络自提出后被广泛应用于图像与视频领域，并在这些领域中的表现达

到了前所未有的高度。卷积神经网络一般由卷积层与池化层构成，卷积层负责从输入数据中提取平面元素的相互特征；池化层负责对数据进行降采样。

此处以二维输入数据举例，卷积的计算公式为：

$$S(m,n) = (I * K)_{m,n} = \sum_{i=1}^{m}\sum_{j=1}^{n} I(i,j)K(m-i,n-j) \tag{4.9}$$

其中$S(m,n)$代表(m,n)处的计算值，I代表输入值，K代表卷积核。卷积运算的过程如图所示：

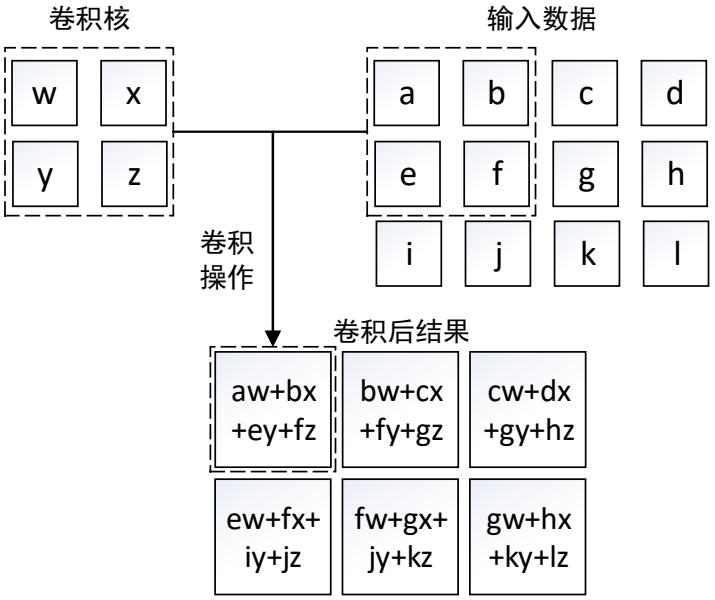

图 4.4 卷积操作过程

Fig. 4.4 Convolution operation

池化主要分为均值池化与最大池化，其计算过程是在某一选定的邻域内计算均值或最大值后，将此邻域使用计算得到的值代替，所以主要作用为特征的降采样，也就是降维。

卷积神经网络主要有三个特征，第一是稀疏连接：在上文提到的全连接神经网络中后一层的神经元与前一层的神经元两两之间均有相互连接，而卷积层则采用卷积核的滑动计算与前一层连接，使得计算量相比于全连接神经网络有所减少；第二是参数共享：在全连接神经网络中，权重矩阵的每个元素只会被使用一次，而在卷积运算中，通过同一个卷积核的滑动来完成一层的运算，这样的计算过程有利于模型学习同一层的不同位置的相互关系，也降低了模型的参数量；第三是等变表示，等变表示是指当多维输入数据发生平移或旋转之后，模型仍然可以学习到输入数据的特征。

4.2.2 文本 CNN

通常 CNN 结构仅被使用在图像与视频领域，而 Kim[40]的研究开创性的指出，对普

通 CNN 稍作修改就可以使用于文本分类领域，其所提出的模型被称之为文本 CNN（Text CNN）。文本 CNN 的结构如图 4.5[40]所示，其可以直接对整个句子进行处理。

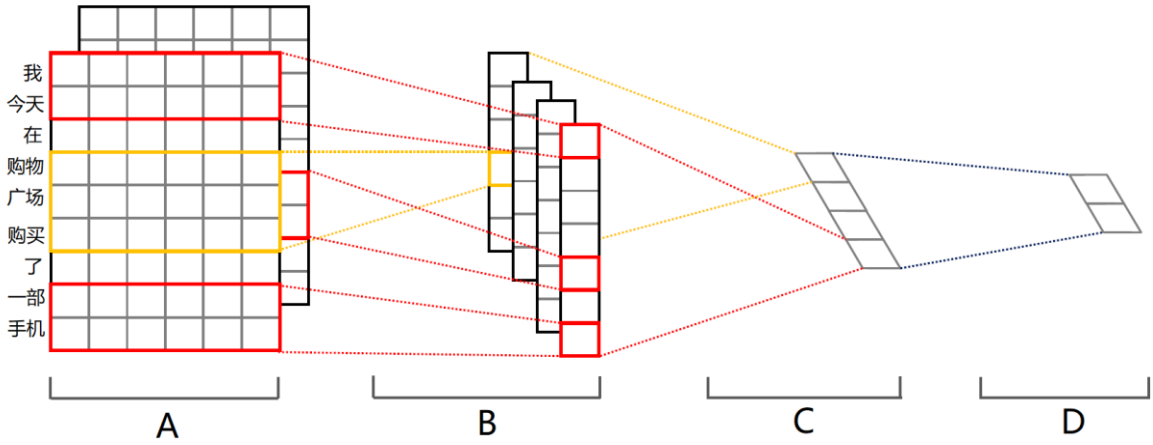

图 4.5 文本 CNN 的结构

Fig. 4.5 Structure of Text CNN

图中的结构主要分为四个部分，第一部分为模型的表示，将句子表示为句中的词向量组成的序列，所以其两个维度分别为句子长度与词向量维度，另外，如果有多种词向量表示方式，可以将其堆叠组成一个三维句子表示。第二部分为卷积处理，图中不同的颜色代表不同的卷积核，采用一组以句子长度做滑动窗口的卷积核来生成不同的特征向量，最终可以得到与卷积核数目相同数目的特征向量。第三步为最大池化操作，在同一特征向量中取出最大的值，并合并为一个新的特征向量。第四部分为模型的输出部分，首先采用全连接神经网络将特征向量维度降低至分类任务的类别数，此处为 2，然后进行 Softmax 处理，并取最大值所在位作为模型的类别输出。

文本 CNN 结构简单，但如果有需要也可以在模型中多次进行卷积与池化以提高模型的复杂度。文本 CNN 在多个公开数据集中都达到了较高的表现。另外，由于 CNN 的计算可以并行，所以文本 CNN 的速度大幅优于基于循环结构的神经网络。

4.3 循环神经网络与高级循环单元

4.3.1 循环神经网络

循环神经网络[41]（Recurrent Neural Network，简称 RNN），是一类专门用于处理序列类型数据的神经网络结构，序列型数据通常是一个具有时间先后或逻辑先后顺序的一系列数据。与卷积神经网络对同一层中的数据采用同一个卷积核的思想一致，循环神经网络结构在序列维度上采用相同的权重，这使得模型可以选择性的保存序列某些位置的信息。另外，循环神经网络还会将上一时刻的隐藏层的值输入到下一时刻进行处理。图

4.6 代表循环神经网络的两种表示方法，其中左侧为折叠式表示方法，右侧为展开式表示方法，A为一个含有激活函数的单层神经网络，x_t与h_t分别表示t时刻的输入与输出。

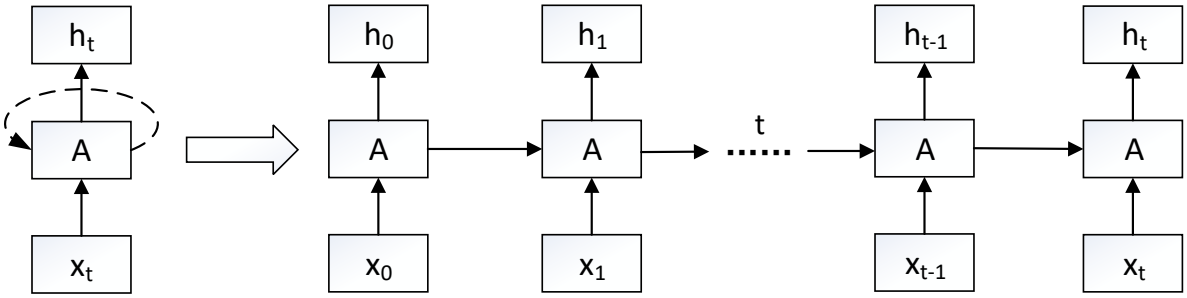

图 4.6 循环神经网络

Fig. 4.6 Recurrent Neural Network

循环神经网络在每个时刻的计算公式如下：

$$a_t = Wa_{t-1} + Ux_t + b \tag{4.10}$$

$$h_t = V\tanh(a_t) + c \tag{4.11}$$

其中a_t代表t时刻隐藏层的值，U,V,W均为权重矩阵，b,c为偏置项，$tanh$代表激活函数，权重与偏置在每个时刻是共用的。

虽然循环神经网络的提出对解决序列问题具有十分重要的意义，但 RNN 有一个严重的缺陷，也就当序列长度较大时，由于循环神经网络中不同时刻的计算单元采用权重共用，导致了在梯度反向传播阶段梯度倾向于消失，所以循环神经网络无法处理序列长度较大的数据。在t时刻隐藏层与初始隐藏层之间的关系为：

$$a_t = (W^t)^T a_0 \tag{4.12}$$

由于权重矩阵W进行了t次自乘，所以对上式进行特征分解之后分析可得，若矩阵W的特征值幅值小于 1，则对应维度的隐藏层的值将会衰减到 0。

4.3.2 长短期记忆单元与门控循环单元

为了解决循环神经网络中梯度消失的问题，Hochreiter 等人[42]研究并发表了长短期记忆（Long Short Term Memory，简称 LSTM）单元，其中对循环神经网络中的基本单元最大的改进为对每个时刻的单元的循环增加了相加权重，并且模型可以基于上下文自行学习权重值，在这种情况下，LSTM 可以对序列中的所有时刻进行决定其是否被学习或遗忘部分或全部上文所传递的计算单元的值，使得长期学习变为可能。

图 4.7 展示了 LSTM 的结构，其中S代表 Sigmoid 激活函数，$tanh$代表 tanh 激活函数，圆圈代表加法与乘法。LSTM 单元引入了三个门控值，门控值的取值范围为[0,1]，其主要功能为控制当前时刻输入与上一时刻计算单元值在一些计算场景中的所占比例。

相对于循环神经网络中不同时刻之间的单个传递值，也就是隐藏层的值，LSTM 单元加入了一个新的值，称为单元状态值（Cell State），单元状态值负责维持不同时刻之间的信息传输，而隐藏层值的作用变为了计算门控值。

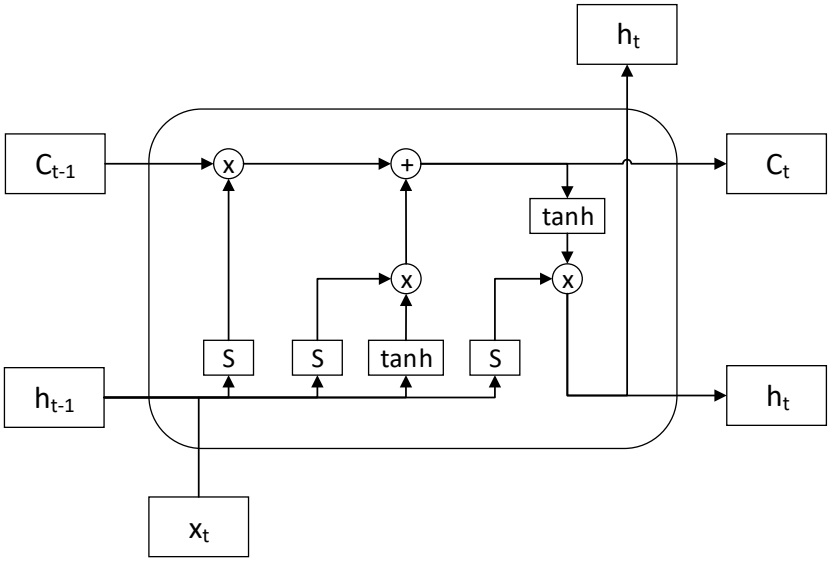

图 4.7 LSTM 单元结构

Fig. 4.7 LSTM cell structure

LSTM 中的门控值计算公式为：

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4.13}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4.14}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{4.15}$$

其中W_*与b_*分别代表不同控制门对应的权重矩阵与偏置，f_t, i_t, o_t分别代表t时刻的遗忘控制门，输入控制门，输出控制门，由于要控制其取值在$[0,1]$范围内，所以这三个值的计算采用的激活函数为 Sigmoid 函数。单元状态值与隐藏层值的计算公式为：

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{4.16}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4.17}$$

$$h_t = o_t * \tanh(C_t) \tag{4.18}$$

其中C_t与h_t分别为单元状态值与，$\tilde{C}_t$用来参与C_t的计算。

LSTM 单元解决了长期依赖问题，并在多个领域尤其是自然语言处理领域取得了巨大的成功。LSTM 结构存在的主要缺陷是由于内部结构较为复杂造成的计算耗时较多，为此，Cho 等人[43]提出了门控循环单元（Gated Recurrent Unit，简称 GRU），相比于 LSTM，GRU 去除了单元状态，并由此减少了一个控制门，另外还去除了控制门计算中的偏置项。GRU 中控制门的计算公式为：

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{4.19}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{4.20}$$

z_t, r_t代表门控值，用于计算隐藏层值。隐藏层值的计算公式如下：

$$\tilde{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{4.21}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{4.22}$$

其中h_t代表隐藏层值。可以看到，GRU 采用$(1 - z_t)$代替了一个门控值，从而将门控值数量减少了一个。GRU 的结构如图 4.8 所示，图中的模块的功能与 LSTM 中相同。

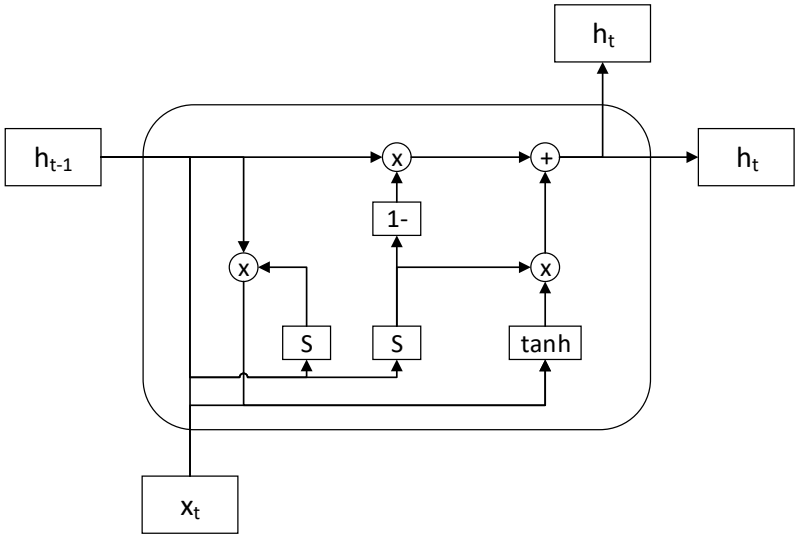

图 4.8 GRU 单元

Fig. 4.8 GRU cell structure

4.3.3 双向 LSTM 结构

循环神经网络结构的先验为，在t时刻的状态只与过去的序列$x_1, \cdots, x_{t-1}$以及当前的输入x_t有关。但在许多应用中，在t时刻的状态可能会与整个输入序列都有关，例如在自然语言处理中，可能会有定语出现在名词的后面的情况，这种情况下在处理t时刻的状态时就需要t时刻后面的信息。

为了解决这个问题，Schuster 等人[44]提出了双向循环神经网络，相比于传统循环神经网络，双向循环神经网络结构在需要双向信息的应用中取得了大幅的表现提高。顾名思义，双向循环神经网络是由两条传统循环神经网络堆叠而成，其中正向循环神经网络负责从前向后处理序列，反向循环神经网络负责从后向前处理序列，两者相互独立，之间没有连接或权重共用。两条循环神经网络共同获取输入数据，并将输出数据进行连接，所以双向循环神经网络的输出维度单向循环神经网络的输出维度的二倍。图 4.9 为双向循环神经网络的示意图，其中s_1至s_m对应每一时刻的输入值，fh_1, bh_1至fh_m, bh_m代表前向与后向各时刻的计算单元，fo_1, bo_1至fo_m, bo_m分别代表各个时刻前向与后向的计算单

元输出值，将两个输出值连接得到最终的输出值。将双向循环神经网络中的循环单元替换成 LSTM 单元，即得到双向 LSTM。

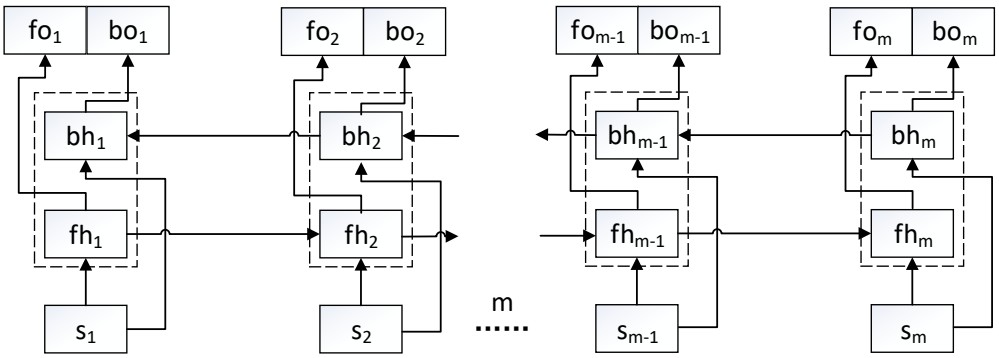

图 4.9 双向循环神经网络

Fig. 4.9 Bidirectional recurrent neural network

在采用单向或双向 LSTM 神经网络对模型进行建模后，为了能使模型输出预测结果，需要再采用类似文本CNN模型中的尾部的结构对LSTM模型输出的结果进行处理。这里假设采用双向 LSTM 神经网络，较为常用的手段有两种，第一种是将正向层与反向层的最终时态的计算单元中的隐藏层值取出并连接，作为双向 LSTM 模型对句子进行编码之后所得的特征向量；第二种方式是将每个时刻的双向 LSTM 的输出组成一个特征矩阵，之后以时间维度做最大池化或均值池化后，得到特征向量。由于第一种方法选择采用最后时刻的隐藏层值，相对于第二种方法其损失的信息更多，所以通常情况下第二种方法较好。两种方法所得的特征向量维度相同，但两种特征生成方法均舍弃了原文中的顺序信息。

4.3.4 自注意力机制

为了弥补以上两种方法的缺陷，Lin 等人[45]研究并提出了一种基于 LSTM 神经网络的特征组合方式，称为自注意力（Self-Attentive）机制。自注意力机制基于上文提及的第二种特征组合方式，并将均值池化修改为加权求和，每一个时刻对应着一个取值范围为[0,1]的实数权重；模型额外定义了一个单层神经网络，该层与每一个时刻的计算单元输出向量相乘并得到一个值，该数值就是本时刻的权重值，所有时刻的权重和为 1。使用每个时刻的权重与此时刻的输出向量相乘之后，对所有时刻所得的结果进行求和，就可以得到最终的特征向量。

自注意力机制的优点在于，模型可以通过额外添加的单层神经网络来控制每个时刻的权重，进而控制在模型生成决策时每个时刻采用了多少信息。相比于直接取最后时刻的隐藏层值以及采用均值池化最大池化等策略，自注意力机制没有丢失输入数据的时间

顺序，所以在多数应用场景，包括情感分析中，均达到了接近最好的效果。

图 4.10 展示了基于双向 LSTM 结构的自注意力机制的结构，其中s_1至s_m为句中的词对应的词向量，fh_1, bh_1至fh_m, bh_m代表前向与后向各时刻的 LSTM 计算单元，fo_1, bo_1至fo_m, bo_m分别代表各个时刻前向与后向的计算单元输出值，u_s代表用于计算权重值的额外单层神经网络，a_1至a_m为计算所得的每一个时刻的权重值，v代表加权求和的结果。

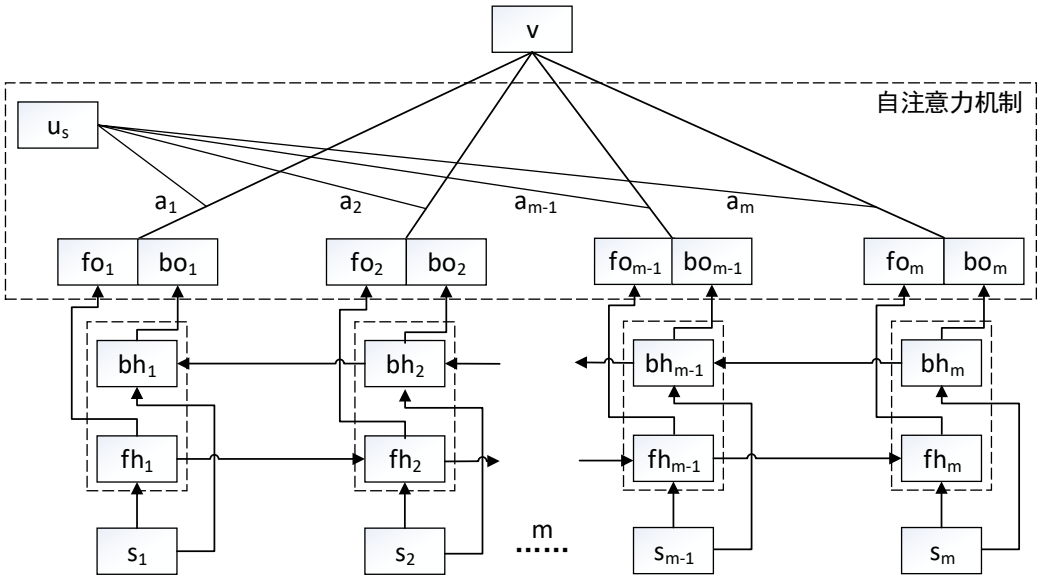

图 4.10 自注意力机制

Fig. 4.10 Self-attention mechanism

4.4 金融文本模型的建立

4.4.1 采用分层注意力网络学习文本表示

分层注意力网络（Hierarchical Attention Networks，简称 HAN）由 Yang 等人[46]研究并提出，该文的研究背景为文本情感分类。模型提出的目的在于，对于某些超长文本，采用普通的双向 LSTM 难以进行建模。作者指出首先对长文本进行切分，之后将两个自注意力机制的双向 LSTM 进行堆叠，底层部分负责对每个文本段中的词进行建模，高层部分负责对底层部分得到的段落表示向量进行建模。分层注意力网络对大量文本的建模问题提出了有效解决方案。

由于本文数据单元中的文本数据的格式为一组并列的等长文本，与分层注意力网络的结构十分符合，所以采用分层注意力网络作为金融文本模型的原始模型进行研究。本章主要讨论文本模型结构，所以此处不将数值数据纳入模型的一部分，并且仅基于包含一个数据单元，即一天数据的数据集进行讨论，并假定其他天数的数据集也符合本节所得到的结论。本文中所采用的分层注意力网络如图 4.11 所示，可以看出该

结构为图 4.10 中所示的自注意力双向 LSTM 结构堆叠而成，图中的标记与 4.10 中类似，此处不再赘述。应注意图中的上下两层虽然结构相同，但在训练过程中并非同一个对象，也就是说两个模块之间没有权重共用的关系。

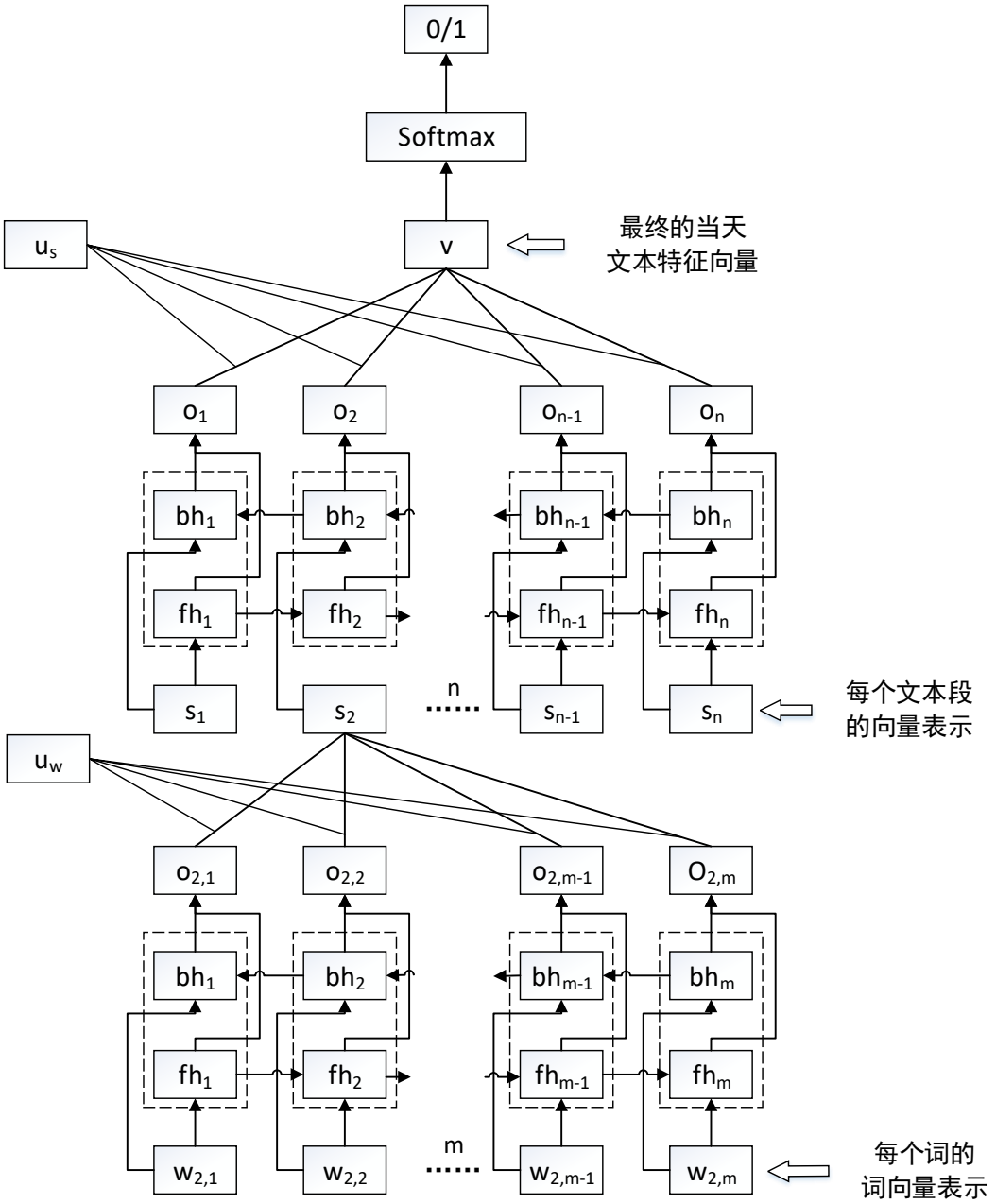

图 4.11 分层注意力网络模型

Fig. 4.11 Hierarchical attention network model

4.4.2 分层注意力网络的缺陷与改进

本文基于一日数据采用上述模型进行训练，以讨论分层注意力网络在本文中的表现，实验条件如表 4.1 所示。

表 4.1 实验条件
Table 4.1 Experimental conditions

操作系统	编程语言	深度学习框架	显卡型号
Windows 10	Python 3.6	Tensorflow 1.8.0	GTX1060-6G

实验中将数据集划分为训练集与测试集，前者占 80%，后者占 20%，使用训练集进行训练，以训练得到的模型在测试集上的表现衡量模型的性能。本节的实验结果仅用于讨论模型结构，并不能代表最终预测模型的表现。实验中所采用的主要参数如表 4.2 中所示。

表 4.2 分层注意力网络的模型参数
Table 4.2 Model parameters for hierarchical attention network

底层时刻数	上层时刻数	批大小	词向量维度	LSTM 维度	特征向量维度	学习率
128	64	32	300	300	1200	0.0003

表 4.3 展示了本次实验结果，表中的“模型结构”一栏进行了简写，代表由两个 LSTM 模块堆叠而成。实验采用提早停止机制作为终止准则，最终进行至 19000 步左右停止，即约进行了 3.7 个 Epoch，一个 Epoch 指对所有的训练数据进行了一轮训练。实验中单个 Epoch 耗 78 时约 78 分钟，总计约 4.9 小时，时间开销过高，若采用 3 日数据或 5 日数据进行预测，则训练过程会非常漫长，使得整个模型的可用性降低；推断耗时是指一个训练成功的模型针对测试数据给出预测结果的耗时，在本次实验中模型对 1000 条样本并行推断的耗时为 18 秒，表现差强人意。最终测试集准确率约为 54.65%。

表 4.3 模型表现与运行时间
Table 4.3 Model representation and running time

模型结构	单个 Epoch 训练耗时	1000 条样本的推断耗时	测试集准确率
LSTM-LSTM	约 78 分钟	约 18 秒	53.65%

分层注意力网络训练速度较慢的原因在于，由于循环神经网络的结构的特殊性，当运行到某时刻的计算单元时，要求本时刻之前的所有时刻均已计算完成，这种性质导致了采用循环神经网络结构的模型训练速度与推断速度缓慢，包括单/双向 LSTM 与 GRU 模型。另外，由于 LSTM 单元中包含 5 个非线性运算模块，与计算 5 个含有非线性激活函数的全连接神经网络耗时相当，所以时间开销极大。

事实上，由于底层负责处理不同文本段的 LSTM 模块之间互不相关，所以理论上可以通过多显卡并行计算来加快训练速度。由于本课题不具备上述硬件条件，所以只能通过修改模型进行加速。

4.4.3 基于自注意力机制的双向 LSTM 与 CNN 分层模型

由于 CNN 中的卷积模块以并行方式进行计算，所以针对 LSTM 串行运算的特点而导致计算缓慢的问题，本文提出采用 CNN 取代分层注意力网络中的部分或全部 LSTM 模块。由于文本段之间为并列关系，并不像一个文本段中的词语之间具有时序或逻辑关系，所以自然地想到将上层 LSTM 更换为 CNN 结构。由于自注意力机制是一种很好的降维方式，在文本处理中自注意力机制与如均值池化、最大池化等方法相比表现结果要好，并且由于采用并行计算其计算耗时不大，所以在上层 CNN 处理之后仍然保留了自注意力机制作为降维方法。本文最终的文本模型如图 4.12 所示，本文将此模型称为基于自注意力机制的双向 LSTM 与 CNN 分层模型，简称 LSTM-CNN 分层模型。

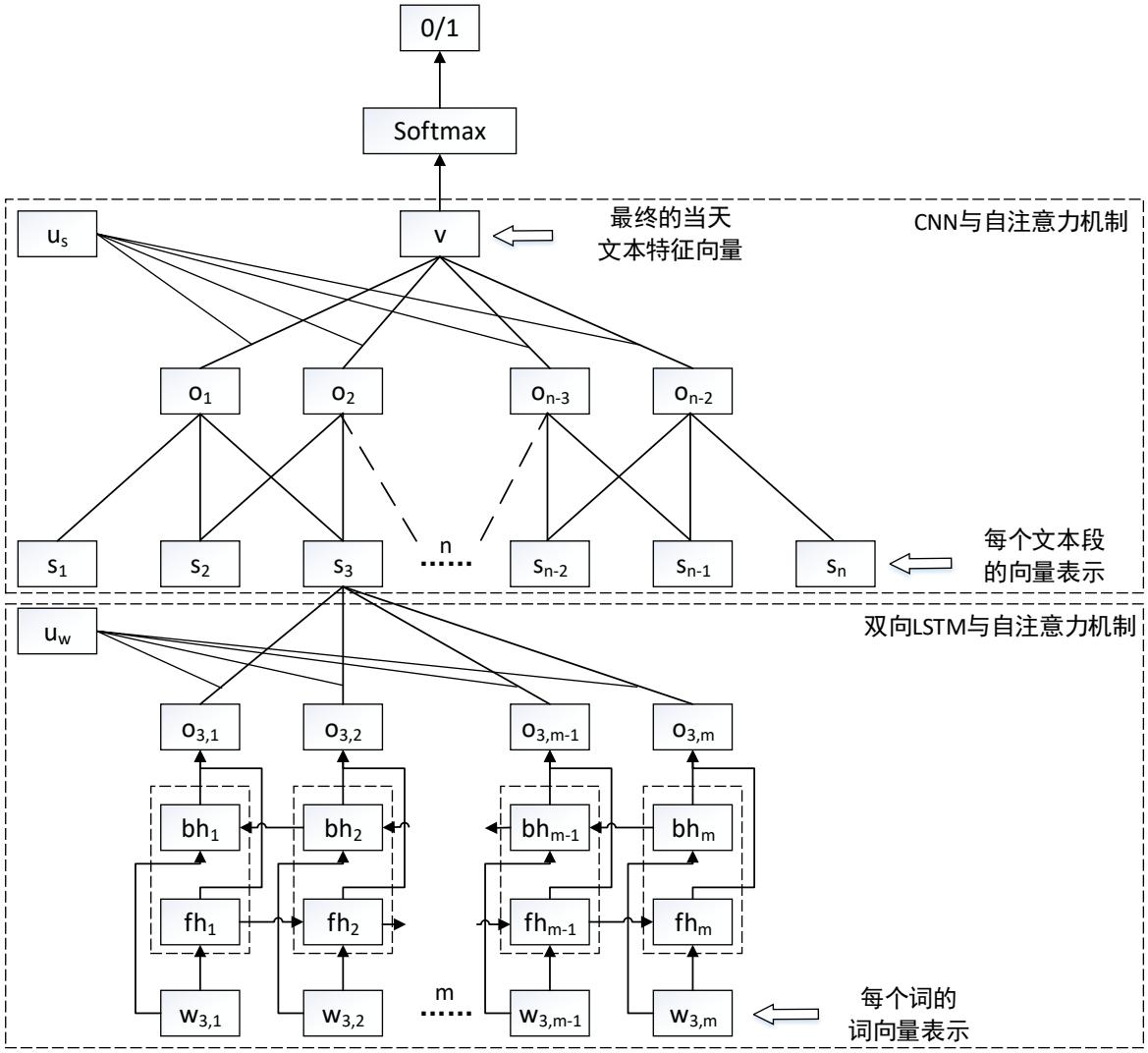

图 4.12 LSTM-CNN 分层模型

Fig. 4.12 LSTM-CNN hierarchical model

图中的 CNN 部分的卷积核宽度为 3，模型采用的参数如表 4.4 所示，卷积核数量为 128 个，所以最终的特征向量维度为 128；卷积过程为单向滑动，并且两侧不进行 0 向

量填补。

表 4.4 LSTM-CNN 的模型参数
Table 4.4 Model parameters of LSTM-CNN

卷积核大小	卷积核个数	批大小	LSTM 维度	特征向量维度	学习率
3*600	128	32	300	128	0.0003

本模型基于一日数据的实验结果如表 4.5 所示。由于 CNN 模块与双向 LSTM、双向 GRU 模块之间可互为替换，所以为了比较所有模型组合并找到最佳模型，本部分共计进行了四个针对不同模型的实验。首先如表中所示，第一行的 LSTM-LSTM 模型为原始的分层注意力网络，其实验结果在上一小节已经进行了分析；第二行的 GRU-GRU 模型仍保留了分层注意力网络的整体结构，但将其中的 LSTM 单元更换为 GRU 单元，本文以此实验来分析 GRU 单元与 LSTM 单元之间的表现差异，由实验结果展示了内部复杂度较低的 GRU 单元对于提高训练速度有一定的帮助，但代价是牺牲了一部分准确率；表中第三行的 LSTM-CNN 为本小节提出的模型，代表底层文本段采用双向 LSTM 进行编码，上层文本段集合采用 CNN 编码，其准确率相比分层注意力网络稍低，但降低了约 35%的训练时间；表中第四行的 CNN-LSTM 作为对照试验之一，采用 CNN 进行底层文本编码，并采用双向 LSTM 编码文本段集合，训练耗时进一步降低，但模型准确率也显著降低；最后一行的 CNN-CNN 采用两个 CNN 模块堆叠构成，训练耗时最短，但准确率也同样最低。综上所述，LSTM-CNN 模型在大幅降低训练时间与推断时间的情况下，其准确率降低幅度在可接受范围内，所以最终选取含自注意力的双向 LSTM 与 CNN 分层模型作为本文中所采用的金融文本模型的基础。

表 4.5 模型表现与运行时间
Table 4.5 Model representation and running time

模型结构	单个 Epoch 训练耗时	1000 条样本的推断耗时	测试集准确率
LSTM-LSTM	约 78 分钟	约 18 秒	54.65%
GRU-GRU	约 70 分钟	约 16 秒	54.21%
LSTM-CNN	约 55 分钟	约 11 秒	54.51%
CNN-LSTM	约 43 分钟	约 8 秒	54.02%
CNN-CNN	约 37 分钟	约 7 秒	53.39%

4.4.4 改进的自注意力机制

本文提出了一种对自注意力机制改进的方法，改进后的结构如图 4.13 所示。在原

来的方法中，原向量值s经过两个方向不同的 LSTM 单元处理后分别得到了fo与bo，如图中两个虚线方框所示，这样的操作损失了原向量值所携带的信息；本文提出对处理后得到的特征向量进行补充，具体方法为将每个时刻的原向量s跳过 LSTM 单元直接与每个时刻的特征向量连接，基于连接后的向量进行后续处理，如计算注意力权重后加权求和等。

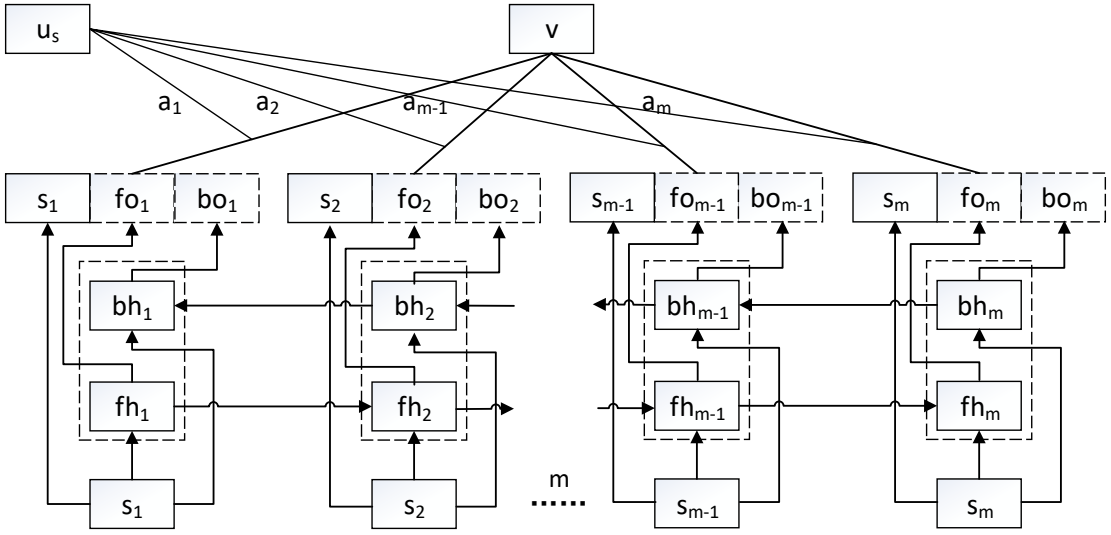

图 4.13 改进的自注意力机制

Fig. 4.13 Improved self-attention mechanism

表 4.6 中展示了与改进前模型的对比结果，改进后的模型在保持训练耗时几乎不变的情况下提升了模型的准确率。

表 4.6 改进的自注意力机制表现

Table 4.6 Improved self-attention mechanism performance

模型结构	单个 Epoch 训练耗时	1000 条样本的推断耗时	准确率
LSTM-CNN	约 55 分钟	约 11 秒	54.51%
改进自注意力的 LSTM-CNN	约 56 分钟	约 11 秒	54.96%

4.5 本章小结

本章主要讨论预测模型中的文本模型部分的原理与结构，首先介绍了模型中采用的深度学习模块，包括全连接神经网络，卷积神经网络与循环神经网络，以及一些基于循环神经网络的高级结构；之后本章基于分层注意力网络搭建了原始文本模型，但实验暴露出分层注意力网络运行缓慢的问题。

为了解决速度缓慢而导致模型训练困难的问题，本章基于分层结构设计了一种融合了 CNN 与 LSTM 以及注意力机制的模型，以 CNN 结构替代部分 LSTM 结构来达到提速的目的，该模型称之为基于自注意力机制的双向 LSTM 与 CNN 分层模型。最终本

章的实验结果表明，该模型兼顾了训练速度与测试集准确率。最后，本章还提出了一种对自注意力机制的改进，实验表明改进后的方法在测试集上的表现有所提高。

第 5 章 基于数值数据与舆论文本的金融市场走势预测系统

在经过词向量模型训练与文本模型搭建之后，本章将给出融合文本数据与数值数据的最终的预测模型。首先本章将介绍用于处理数值数据的神经网络结构，该部分结构主要基于全连接神经网络；之后将文本模型与数值模型得到的特征向量连接，作为最终预测模型部分的输入。

由于最终所采用的数据为多天数据，所以模型的最后模块采用了双层堆叠 LSTM 神经网络，并采用高速公路网络降低由于堆叠造成的模型难以学习的问题。另外本章介绍了模型的损失函数与评价指标，以及一些模型训练过程中采用的训练技巧。最终本章在时间维度与数据维度上对模型表现进行了比较，并分析了原因。

5.1 预测系统整体框架

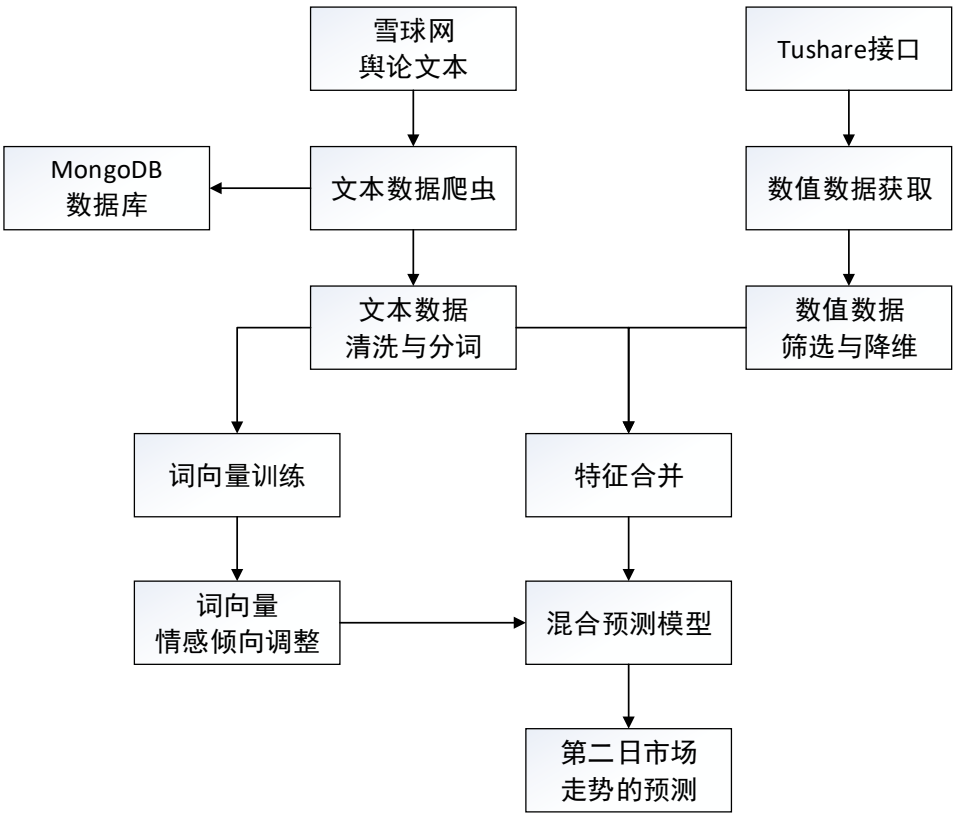

图 5.1 预测系统整体框架

Fig. 5.1 The overall framework of the forecasting system

图 5.1 展示了本文中预测系统的整体框架，整个模型数据的来源由两部分组成，分

别为雪球网市场舆论文本与 Tushare 金融数据接口，其中舆论文本采用爬虫进行采集，数值数据直接调用接口即可获得；在获得两种数据后，对文本数据进行清洗与分词，对数值数据进行分析相关性并降维。

在对两种数据进行获取并处理之后，一方面，将处理后的文本数据进行词向量训练，并基于情感倾向词典对预训练得到的词向量进行调整；另一方面，将文本数据与数值数据相结合构成市场走势的预测数据，并采用调整后的词向量对原文本数据进行映射，将其作为混合预测模型的输入。最终由预测模型给出第二天市场的走势预测。

5.2 走势预测模型

5.2.1 金融数值数据模型

本小节设计对金融数值数据进行处理的模型。如前文所述，数值数据为一个 13 维的向量，无需采用复杂的结构进行编码，所以数值数据采用全连接神经网络进行处理，之后再与文本特征向量合并。原始数据由于量纲不一致，若直接将其放入神经网络则会导致模型训练速度慢并且损失函数难以收敛，所以需要首先对其进行归一化或标准化操作去量纲。

本文选择标准化操作去量纲，并且采用神经网络中的批标准化层[47]（Batch Normalization Layer，简称 BN）结构来实现本步骤。标准化值的计算公式如下：

$$\mu = \frac{1}{m}\sum_i x^{(i)} \tag{5.1}$$

$$\sigma^2 = \frac{1}{m}\sum_i (x_i - \mu)^2 \tag{5.2}$$

$$x_{norm}^{(i)} = \frac{x^{(i)} - \mu}{\sqrt{\sigma^2 + \varepsilon}} \tag{5.3}$$

其中i代表同一批次中不同数据，m代表批次中的数据个数，μ为本批次数据的均值，σ^2为方差，ε是为了防止分母为 0 的值，可取10^{-4}。通过以上的计算，可以将该层的输出值转化为服从 0 均值，1 方差的正态分布。

在完成标准化之后，批标准化层还有一个计算过程，是按照一定比例将标准化后的值进行放缩，放缩比例是一个可学习的参数，这样就使得模型自身可以确定数据放缩的程度。放缩过程的计算公式如下：

$$\tilde{x}^{(i)} = \gamma \cdot x_{norm}^{(i)} + \beta \tag{5.4}$$

其中γ与β为可学习参数，可在模型中通过梯度下降法学习；注意到若使$\gamma = \sqrt{\sigma^2 + \varepsilon}$且$\beta = \mu$，则有$\tilde{x}^{(i)} = x_{norm}^{(i)}$，即标准化值被完全还原。

本文首先将金融数值数据采用批标准化层进行处理，之后通过两层等宽的全连接层处理后，生成数值数据的最终特征向量，特征向量将与文本特征向量一同作用于预测模型。模型结构如图 5.2 所示。

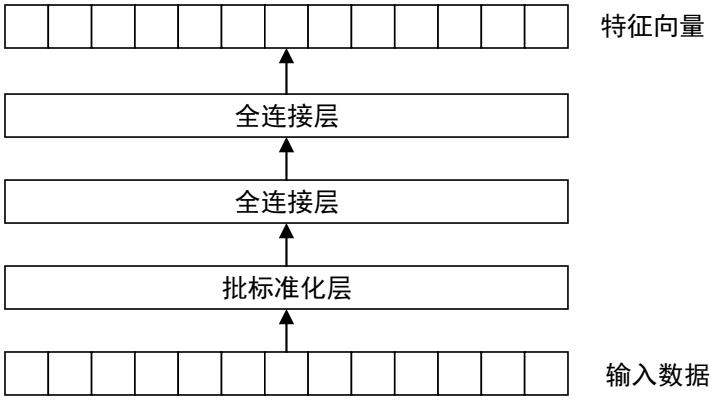

图 5.2 数值数据部分的模型

Fig. 5.2 The model of the numerical data section

5.2.2 高速公路网络模型

由于本文在最终的预测模型中采用了堆叠的循环神经网络，当采用堆叠模型时，模型通常难以训练，从而导致模型最终表现较差。针对这个问题，本文在模型中进入了高速公路网络。高速公路网络（Highway Networks）模型是由 Srivastava 等人[48]研究并发表的一种网络结构，其旨在解决当神经网络的层数过多时模型难以有效训练，从而导致最终模型表现较差的问题。高速公路网络的思路为，额外定义一个单层神经网络，对于某一个神经网络结构，通常是 LSTM 或 CNN，此结构的输出会与单层神经网络相乘得到一个权重，该权重负责控制最终输出值中本结构的输入值与输出值之间相加的比值；将计算得到的最终输出值作为本结构的输出值传递给网络中的下一个计算结构。高速公路网络的计算公式为：

$$y' = H(W_H, x) \tag{5.5}$$

$$\mathrm{c} = \sigma(W_c y' + b_c) \tag{5.6}$$

其中y'为原神经网络结构的输出，x为神经网络结构的输入，$H(\cdot)$代表神经网络操作，如上文所述；c代表门控值，σ代表 Sigmoid 函数，W_c与b_c代表权重矩阵与偏置。经过上述计算之后，最终得到新的输出值为：

$$y = c \cdot y' + (1 - c) \cdot x \tag{5.7}$$

通过添加高速公路网络，可以在提高模型复杂度的同时保证模型的表现得以提高。本文主要将其与循环神经网络结构相结合，由于高速公路网络模块有一定的计算量，所以为了避免大幅提高计算量，没有将该模块加入到文本模型之中。

5.2.3 基于改进的自注意力机制的双向 LSTM 与 CNN 分层混合模型

本文最终的预测模型如图 5.3 所示，其中最底层的模块为一个基于自注意力的双向 LSTM 神经网络，用于对每个文本段进行处理，最终生成每个文本段的向量表示，m代表文本段中词语的个数，此处选取的值为 128。

中下层对文本段向量集合进行处理，采用卷积神经网络进行卷积操作，并对生成的向量组采用自注意力机制降维，其中n为集合中的文本段个数，此处选取的值为 64。该部分与最底层模块共同构成了文本处理模型，该模型已经在第四章讨论完成，所以此处对于这一部分的细节不再进行赘述。中下层的另一部分对数值数据进行处理，该部分的模型在上一小节给出。将得到的文本特征向量与数值特征向量拼接，得到最终当天的混合特征向量。

中上层对的得到每个日期的特征向量进行处理，t代表数据包含的天数，如第二章所述，t的取值可以是 3、5、7，本章会对不同的t取值进行比较。由于不同日期之间的数据满足时序关系，所以本模块选取的模型为一个双层堆叠的双向 LSTM 神经网络，所以共计为 4 层，且正向传递层与反向传递层之间没有连接关系。正向传递的第一层与反向传递层的第一层均采用了高速公路网络进行处理，目的是为了解决由模型堆叠导致的训练困难与效果变差的问题。最终，将每个时刻得到的特征向量进行横向拼接，得到最终特征向量。

最上层对第二天市场的走势给出预测，将最终特征向量采用 Softmax 进行处理后，得到明日走势的类别，即上涨或下跌。模型中各部分的参数如表 5.1 所示（顺序为由图底端至图顶端）。

表 5.1 主要参数值

Table 5.1 Main parameter values

参数名称	参数数值	参数名称	参数数值
词向量维度	300	批正则化层维度	13
文本 LSTM 单元维度	300	全连接神经网络维度	13
文本段词语数	128	文本段注意力层维度	128
文本注意力层维度	256	最终单天特征	141
文本段向量表示	900	数据天数	3,5,7
文本段个数	64	日数据 LSTM 单元维度	141
卷积核尺寸	3*900	最终特征向量维度	141*t
卷积核个数	128	类别数	2

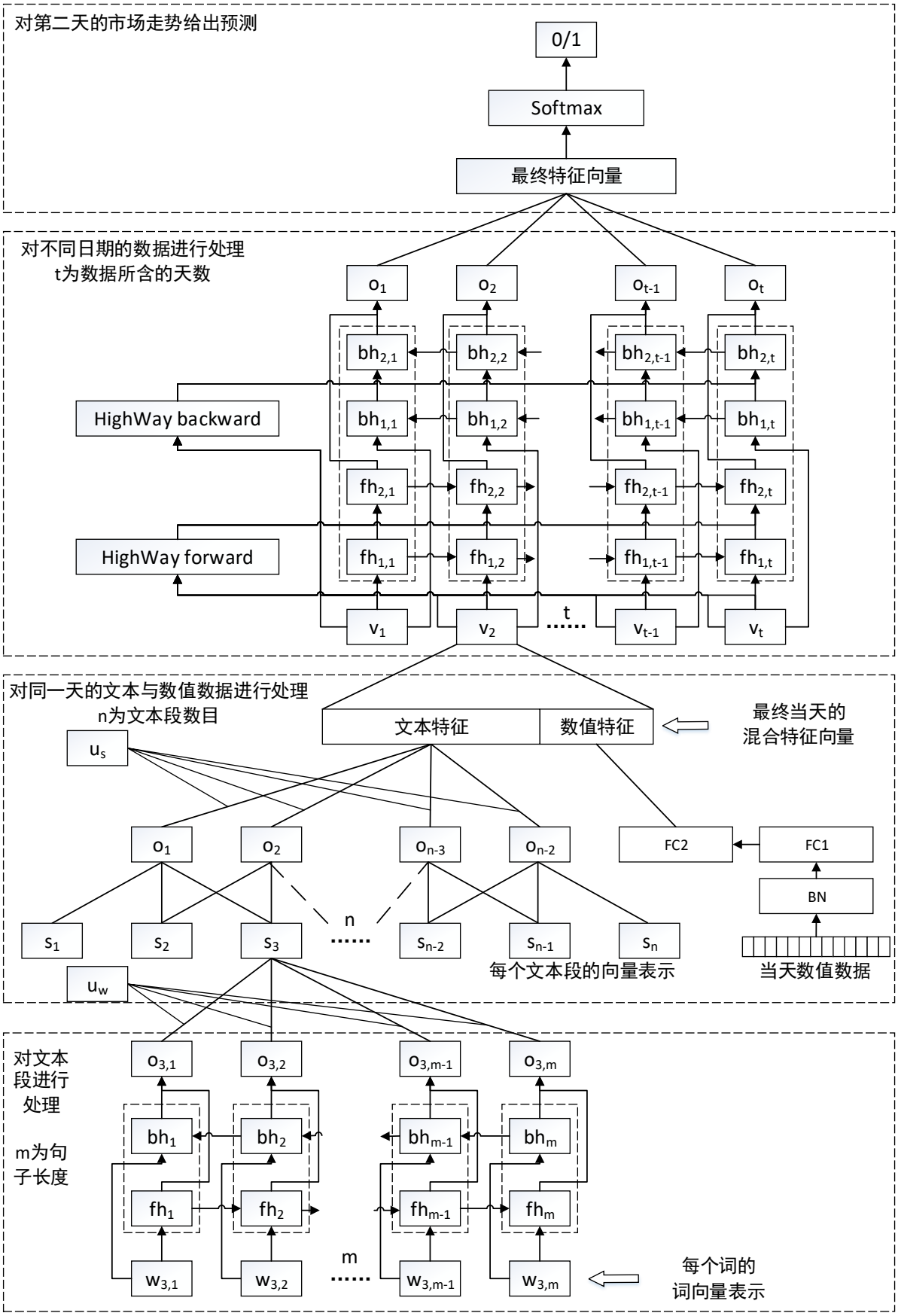

图 5.3 整体预测模型

Fig. 5.3 Overall prediction model

5.3 模型训练技巧与训练过程

5.3.1 实验条件

本章的实验条件与第四章相同，实验条件如表 5.2 所示。本文所采用的深度学习框架为 GPU 版本的 Tensorflow，Tensorflow 是一个由谷歌大脑团队[49-52]开发的开源符号数学库，用于各种机器学习及神经网络算法的编程实现，是目前使用人数最多的深度学习框架。

表 5.2 实验条件

Table 5.2 Experimental conditions

操作系统	编程语言	深度学习框架	显卡型号
Windows 10	Python 3.6	Tensorflow 1.8.0	GTX1060-6G

5.3.2 实验细节与训练技巧

由于本文中的预测任务为二分类任务，所以模型的损失函数采用交叉熵损失函数，损失函数的公式如下：

$$Loss = y log\hat{y} + (1-y)\log(1-\hat{y}) \tag{5.8}$$

其中对数以自然常数e为底，y代表真值，取值为 0 或 1，$\hat{y}$为预测值，数据类型为浮点数，取值范围为[0,1]，$\hat{y}$的值代表当前样本取值为 1 的概率，若$\hat{y}$大于 0.5 则预测结果为 1，若小于 0.5 则预测结果为 0。

由于模型可能存在过拟合的风险，亦即模型在训练结束后在测试机上的表现远远低于在训练集上的表现的情况。为了减轻模型的过拟合风险，需要采取一些防止过拟合的手段，其中较为有效的手段是向损失函数中加入正则化项，本文采用L_2正则化项，正则化项的值为模型中所有权重值的L_2范数的平方。加入了正则化项后的损失函数公式如下：

$$Loss = y log\hat{y} + (1-y)\log(1-\hat{y}) + \alpha\|\omega\|_2^2 \tag{5.9}$$

其中$\alpha\|\omega\|_2^2$为L_2正则化项，α为权重因子，用于控制式中损失函数与正则化值得比例，在本文中的值为 0.001，ω代表模型中的权重值。

本文中的模型评价标准采用二分类正确率与 F1 值进行衡量，其中正确率的定义公式如下：

$$acc = \frac{1}{m}\sum_{i=1}^{m}[\![\hat{c}_i = y_i]\!] \tag{5.10}$$

其中m代表本次预测的样本数量，$\hat{c}_i$代表第i个样本的预测类别，y_i代表第i个样本

的真实类别，⟦·⟧运算符代表当其中的表达式成立时取值为 1，不成立时取值为 0。

F1 值是一种用来衡量二分类模型表现的一种指标，其同时兼顾了模型的准确率与召回率。F1 值的计算公式如下：

$$\mathrm{F1} = \frac{2\times TP}{TOTAL+TP-TN} \tag{5.11}$$

其中$TOTAL$代表样本总数，TP为 Ture Positive 的缩写，代表所有正例样本被预测为正的个数，在本文中正例样本代表上涨类；TN为 Ture Negative 的缩写，代表所有负例样本被预测为负的个数，在本文负例样本中代表下跌类。

在训练过程中，本文还采用了一些技巧来提高模型的表现，以下对主要采取的技巧进行简介：

（1）提前终止（Early Stopping）。提前终止是一种控制模型停止训练的方法。随着训练过程的继续，通常模型在训练集上的训练误差会不断减小，而测试集误差当达到某一个极小点之后会有反弹的倾向，当采用固定训练轮数控制训练过程是，模型通常不会停止在测试集误差极小点处，这样就造成了最终模型处于欠拟合或过拟合状态。提前终止的做法是，若测试集误差在某一段训练过程中没有降低，就将上一次训练集误差降低的时刻所存储的模型作为最终模型，并停止训练过程，本文中设定的时间段为 5000 步。

（2）神经元随机失活（Dropout）。Dropout[53]是一种降低模型过拟合程度的方法，该方法的思想为在每一次计算过程中，按照事先选取的比例随机地将同一层之中的部分神经元失活，与失活神经元相关的任何计算均不会被进行，这样的操作使得模型在计算时，不会过多地依赖于某几个神经元，而是将注意力平均地分散到同一层的神经元之中。本文将 Dropout 应用于仅有的两个全连接层与所有的 LSTM 单元的输出层中，每层有 10%的神经元失活。

（3）词向量微调（Fine-Tune）。词向量微调是指将输入模型的词向量表声明为可学习的参数矩阵，而非不可调整的常数矩阵。词向量被声明为参数矩阵后，模型可根据需要对词向量元素值进行调整，通常可使模型的表现得以提升。该方法的思想来源于迁移学习。

（4）梯度截断（Clipping Gradient）。梯度截断[54]是一种对优化方法的改进。具体来说，在对损失函数的极小化过程中，可能会出现梯度值较大导致更新过程的步幅过大，或损失函数的超曲面存在“悬崖”区域导致无法达到极小点的情况。以上两种异常情况均可采用梯度截断方法来解决。梯度截断的实现方法为对梯度向量逐元素地除以梯度的范数，处理后的梯度即为经过截断的梯度。梯度截断方法与本文所采用的 Adam 优化方法[55]可以有效解决上述两个问题。

5.3.3 模型训练过程

本文在训练过程中采用 TensorBoard 工具监控模型训练过程。TensorBoard 工具为 TensorFlow 团队开发的一个用于分析神经网络模型训练过程与训练结果的工具包，该工具包提供了参数分布统计、计算图生成、关键值监测等功能，本文采用关键值监测功能对训练过程中的训练误差、训练集正确率、测试集正确率进行研究。

图 5.3，图 5.4，图 5.5 分别代表本次实验中的训练误差曲线、训练集正确率曲线与测试集正确率曲线，实验中采用的预测数据时长为 3 日数据，将其中 80%作为训练集，其余 20%作为测试集，在分离数据集时采用分层抽样避免发生类别不均。为了生成稳定的图形，此处采用了固定步数训练方式，当训练至 22000 步时自动停止训练过程，并存储训练集正确率表现最好的时刻的模型。图中折线图的数据点之间的间隔为 100 步，淡色曲线代表实际值曲线，深色曲线代表经过平滑之后的曲线，由于训练过程采用小批量梯度下降法，每次取到的样本不一定相同，所以导致训练误差的真值曲线与训练集正确率的真值曲线相比于测试集正确率波动较大。

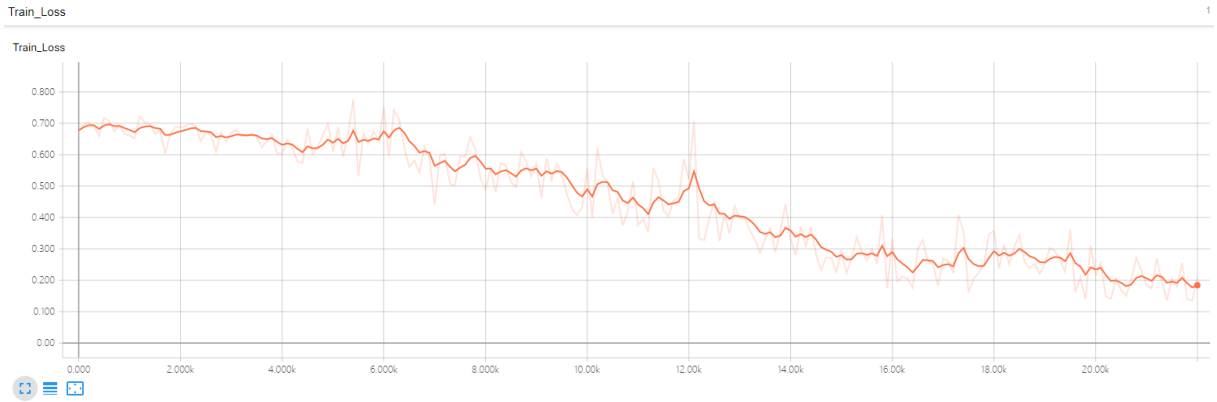

图 5.4 训练误差

Fig. 5.4 The training error

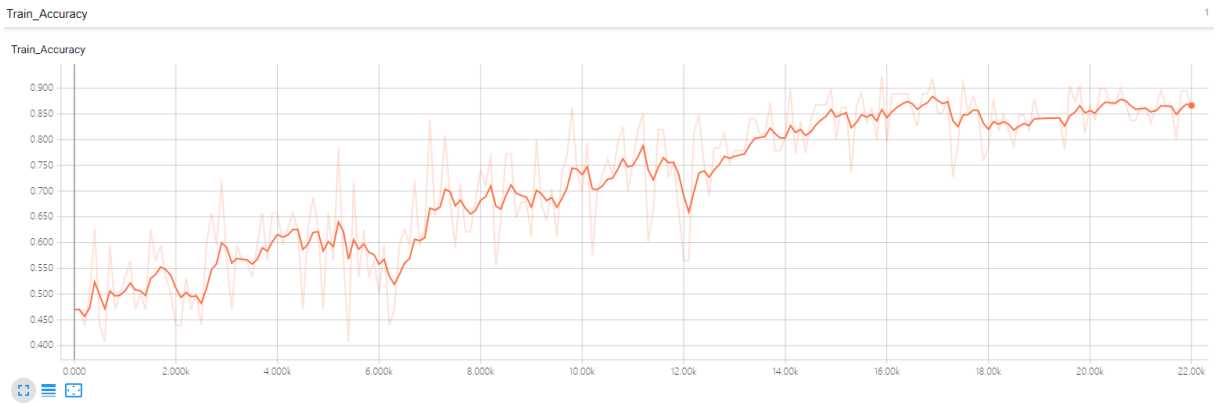

图 5.5 训练集正确率

Fig. 5.5 Accuracy on the training set

观察测试集正确率曲线，测试集正确率约在 17000 至 18000 的训练步数内达到了高峰，约为 58.8%，之后随着训练过程的增加，测试集正确率开始降低，直至训练结束；而在同一时段内，训练误差不断降低，训练集正确率仍然在提升，可以确定在这一时间段内模型处于过拟合状态。

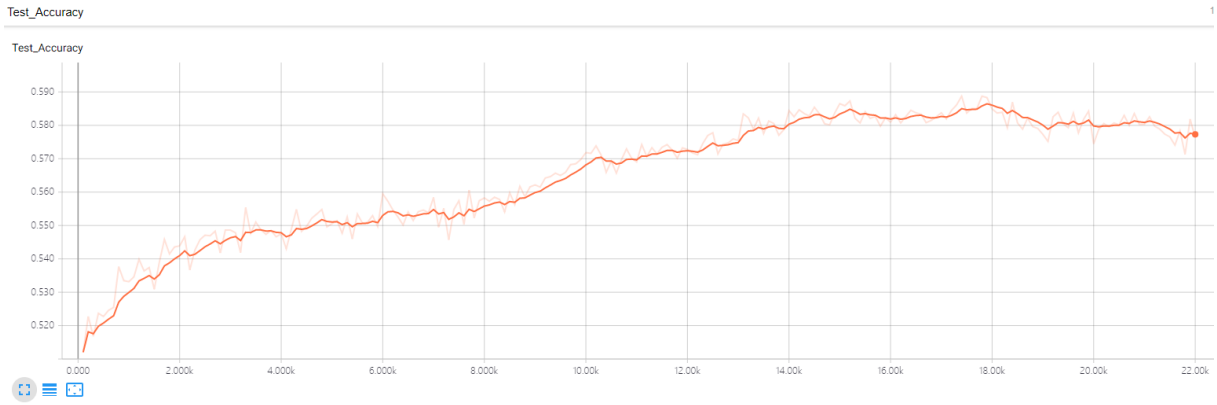

图 5.6 测试集正确率

Fig. 5.6 Accuracy on the test set

5.4 实验结果对比与分析

5.4.1 不同时长数据结果之间的差异

本文在第二章中根据不同的预测时间生成了三种不同的数据集，所依据的时间长度分别为预测前的 1、3、5 天，三类数据的详细统计信息如表 5.3 所示。表中不同天数的数据量与类别比例不同的原因是在生成数据的过程中舍弃了任何在一天中不包含任何文本的数据，当空文本数据单元出现时，时间窗口越大受波及的样本就越多，所以数据时长越长样本数越少。

表 5.3 三类数据的统计特征

Table 5.3 Statistical characteristics of three types of data

数据时长	总样本数	正例占比
前 1 天	203742	50.6%
前 3 天	198549	50.1%
前 5 天	175329	50.4%

由于 1 天时长的数据不存在时间序列，所以 1 天时长采用的模型没有混合模型中的中上层部分，得到当天的组合特征之后就可以给出预测；3 天与 5 天时长的数据采用的是本章的混合模型，最终的时间跨度值t与每类数据的时长相同。为了控制变量，模型中其余参数在训练过程中均保持相同。模型每训练 5000 步就会将学习率减半，并采用提

前终止方法自行决定是否结束训练，判断标准为若测试集正确率在5000步内没有下降，则终止训练，并将 5000 步前时刻存储的模型作为最终模型。另外，实验中采用的词向量均为二阶段训练后的词向量实验结果如表 5.4 所示。

表 5.4 三类数据的实验结果

Table 5.4 Experimental results of three types of data

数据时长	单轮训练耗时	自动停止前训练轮数	测试集正确率	F1 值
前 1 天	约 57 分钟	3.6	56.3%	55.6%
前 3 天	约 83 分钟	4.2	58.8%	56.4%
前 5 天	约 102 分钟	4.1	59.2%	57.3%

由表中结果可以得到，随着数据时长的增加，单轮训练耗时几乎呈线性增长；模型表现方面，由 5 天数据训练所得的正确率表现最好，但训练耗时也最高，3 天数据与 5 天数据在正确率方面相差不大；1 天时长数据的表现明显低于其他两种时长的表现，且训练步数也较低，说明对于预测模型，时长较短的数据所含信息相对不足。

由于模型采用了提前终止方法，所以模型可能会对测试集产生过拟合。实验结果中的 F1 值与正确率之间的差值大小有助于分析模型的过拟合程度，在二分类样本类别较为均衡的情况下，若 F1 值与正确率值差距较大，表明模型在测试阶段产生了对测试集的过拟合现象。本轮实验中 F1 值与正确率的差值不大，表明模型对测试集不存在或存在轻微的过拟合。

5.4.2 词向量二阶段训练对预测结果的影响

由于采用传统词向量预训练算法得到的词向量无法有效地表示词语的情感极性，本文在第三章中采用词向量二阶段训练方法对词向量进行二次训练。本小节对针对两种不同的词向量分别进行实验，并比较两次实验的表现差异。实验采用的数据时长为三天数据，实验结果如表 5.5 所示。

表 5.5 两种词向量的实验结果

Table 5.5 Experimental results of two word vectors

词向量类型	单轮训练耗时	自动停止前训练轮数	测试集正确率	F1 值
预训练词向量	约 83 分钟	4.7	57.2%	55.7%
二阶段训练词向量	约 83 分钟	4.2	58.8%	56.4%

由实验结果可以看出，相比于预训练词向量，二阶段训练后的词向量提高了测试集正确率与 F1 值，并降低了训练时间。实验证明了二阶段词向量训练的有效性。

5.4.3 文本数据与数值数据对预测结果的影响

为了比较文本数据与数值数据中哪种数据对模型的预测结果影响较大，本小节在三日数据的基础上，仅采用文本数据与数值数据进行了实验。当仅采用文本数据时，将混合模型中的数值数据模型去除，并依次减小受影响的位置的神经网络维度；采用数值型数据时，将混合模型中的文本数据模型去除，删除部分包括模型底层与中下层中的部分结构。实验结果如表 5.6 所示。

表 5.6 单类型数据的实验结果

Table 5.6 Experimental results of single type data

数据类型	单轮训练耗时	自动停止前训练轮数	测试集正确率	F1 值
所有	约 83 分钟	4.2	58.8%	56.4%
仅文本	约 78 分钟	3.7	56.1%	54.8%
仅数值	约 10 分钟	1.8	52.2%	52.5%

实验结果表明，仅采用文本的预测模型表现要优于数值模型数据，且模型在仅使用数值数据时训练步数明显低于其他两种情况，说明数值数据对预测模型提供的信息少于文本数据。

5.4.4 稀疏文本对预测结果的影响

由于市场中各个股票的受关注度不一致，关注度较低的股票的舆论文本也较少，亦即存在一部分股票的每日文本数少于 64 条。对于这些情况，本文在处理时会采用 0 向量进行补齐至 64 条文本，这样就造成了文本数据稀疏的问题。由于对稀疏数据进行处理相当于对训练过程引入了噪声，所以通常情况下稀疏数据会严重降低模型的表现。本小节将对本文中稀疏数据对预测模型表现的影响进行分析。

本轮实验基于 3 日数据进行。首先对文本数据进行过滤，过滤规则为若当天文本数未达到容量的 50%，即 32 条，则所有包含当日的数据将被舍弃。使用过滤后的数据进行训练得到的结果如表 5.7 所示。

表 5.7 过滤后数据的实验结果

Table 5.7 Experimental results of filtered data

数据类型	总样本数	正例占比数	测试集正确率	F1 值
过滤前	198549	50.1%	58.8%	56.4%
过滤后	63427	50.8%	62.7%	63.1%

对文本数据进行过滤后，约有三分之二的样本被舍弃。实验结果表明，采用过滤后

的数据后，模型的表现相比于过滤前的数据有很大提升，亦即稀疏文本数据对模型的表现产生了较大的影响。

5.5 本章小结

本章内容为给出了最终的混合预测模型以及进行了多组实验对比某些变量对模型表现的影响。首先本章给出了处理数值数据的模型部分，并将其与上一章获得的文本模型相结合，最终得到混合预测模型，本文中称其为基于改进的自注意力机制的双向 LSTM 与 CNN 分层混合模型。

在基于一些训练技巧进行模型训练后获得的对比实验结果表明，模型在 5 日数据上的表现最好，二分类正确率为 59.2%，亦即模型所采用的数据时间段越长越好，但数据时间越长训练过程也越慢；本章的实验证明了词向量二阶段训练可以提高模型的测试集正确率；本章还探讨了不同类型数据对预测模型的决策的贡献程度，实验结果表明文本数据向预测模型提供的信息远多于数值数据；最后，本章探讨了稀疏文本数据对预测模型表现的影响，实验结果表明，在采用去除稀疏数据后的文本进行预测时，模型表现明显提升。

第 6 章 总结与展望

6.1 总结

随着国内金融市场的受关注程度越来越高，投资者也越来越多，准确地把握市场走向，有利于监管部分及时作出响应并颁布相应政策，也有利于指导个人投资者做出投资决策。传统的走势预测模型主要依靠价格数据、基本面数据等数值数据进行预测，而近年来深度学习与自然语言处理技术的发展，以及在线社交平台的兴起，均为市场走势预测领域带来了新的研究机遇。

本文基于深度学习与自然语言处理技术对股票市场的走势预测问题进行了研究，本文主要研究工作与主要研究成果如下：

（1）对金融舆论文本数据与数值数据进行采集与处理，并进行了必要的统计分析。本文采集了雪球网的投资者讨论与公司发布等文本数据，进行了统计分析并发现了文本数据的一些特征；另外对文本数据进行了数据预处理并对数值数据进行了相关性分析与降维。

（2）基于金融文本数据训练了一个经过调整的词向量模型。首先基于文本数据进行了词向量模型的初次训练，为了解决词向量模型在情感分析领域存在的弊端，需要对初次得到的词向量模型进行二阶段训练。本文对前人的研究进行了改进，并基于金融领域词典对词向量模型进行了二阶段训练。

（3）提出了一个适用于本文研究背景的深度学习文本模型。本文针对研究中采集到的金融舆论文本数据的结构特点，并借鉴了前人的研究思路，提出了一种神经网络结构，称之为基于自注意力机制的双向 LSTM 与 CNN 的分层模型，该模型在实验中表现较好，兼顾了训练速度与精度；另外本文对自注意力机制提出了改进，实验表明改进后模型表现有所提升。

（4）基于两种数据设计了可处理多天输入数据的混合走势预测模型。本文将文本数据处理结构与数值数据处理结构合并，并引入堆叠循环神经网络与高速公路网络处理多天输入的模型结构，该模型称为基于改进的自注意力机制的双向 LSTM 与 CNN 分层混合模型，该模型在基于本文数据集的实验中达到了较高的正确率。

6.2 展望

本文采用爬虫技术与数据库技术采集了相应舆论文本与数值数据，并基于深度学习

与自然语言处理技术设计了一个股票市场走势预测模型，最终取得了较好的预测结果，但本文仍有一些可以深入研究的内容，主要为以下几点：

（1）本文所提出的模型结构复杂度过高，导致训练时间居高不下，在之后的研究中可在保持正确率不降低的情况下，研究部分模块的替换结构以降低模型时间复杂度。

（2）本文没有将新闻数据纳入模型所采用的数据，在之后的研究中可以基于新闻文本数据与金融知识图谱系统对本模型进行扩充，以提高预测正确率。

（3）本文的实践结果表明，稀疏文本数据对模型的预测表现有较大的影响，在之后的研究中可以考虑如何降低稀疏文本对模型的影响。

参考文献

[1] 中国互联网络信息中心, 李静. 第 41 次《中国互联网络发展状况统计报告》发布[J]. 中国广播, 2018(3):96-96.

[2] Box G E P, Jenkins G M. Time Series Analysis: Forecasting and Control[J]. Journal of Time, 2010, 31(4):303-303.

[3] White H. Economic prediction using neural networks: the case of IBM daily stock returns[C]// IEEE International Conference on Neural Networks. IEEE, 1988:451-458 vol.2.

[4] Kimoto T, Asakawa K, Yoda M, et al. Stock market prediction system with modular neural networks[C]// IJCNN International Joint Conference on Neural Networks. IEEE, 1990:1-6 vol.1.

[5] Fernandez-Rodrıguez F, Gonzalez-Martel C, Sosvilla-Rivero S. On the profitability of technical trading rules based on artificial neural networks:: Evidence from the Madrid stock market[J]. Economics letters, 2000, 69(1): 89-94.

[6] 郑建刚, 王行愚, 牛玉刚. 基于改进免疫遗传算法的神经网络及其在股票预测中的应用[J]. 华东理工大学学报, 2006, 32(11):1342-1345.

[7] Kyoung-jae Kim. Financial time series forecasting using support vector machines[J]. Neurocomputing, 2003, 55(1):307-319.

[8] 王波, 张凤玲. 神经网络与时间序列模型在股票预测中的比较[J]. 武汉理工大学学报(信息与管理工程版), 2005, 27(6):69-73.

[9] 秦焱, 朱宏, 李旭伟. 基于改进型粒子群优化算法的 BP 网络在股票预测中的应用[J]. 计算机工程与科学, 2008, 30(4):66-68.

[10] Sui X S, Qi Z Y, Yu D R, et al. A Novel Feature Selection Approach Using Classification Complexity for SVM of Stock Market Trend Prediction[C]// International Conference on Management Science and Engineering. IEEE, 2008:1654-1659.

[11] Zhang J, Cui S, Xu Y, et al. A novel data-driven stock price trend prediction system[J]. Expert Systems with Applications, 2018, 97: 60-69.

[12] 吕涛, 郝泳涛. 基于 K 线序列相似性搜索的股票价格预测[J]. 计算机应用, 2017(a02):229-235.

[13] 吕涛, 郝泳涛. 基于相似性匹配和聚类的 K 线模式可盈利性研究[J]. 计算机科学, 2018, 45(3):182-188.

[14] Ding X, Zhang Y, Liu T, et al. Using Structured Events to Predict Stock Price Movement: An Empirical Investigation[C]// Conference on Empirical Methods in Natural Language Processing. 2014:1415-1425.

[15] Ding X, Zhang Y, Liu T, et al. Deep learning for event-driven stock prediction[C]// International Conference on Artificial Intelligence. AAAI Press, 2015:2327-2333.

[16] Ding X, Zhang Y, Liu T, et al. Knowledge-driven event embedding for stock prediction[C]//Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. 2016: 2133-2142.

[17] Akita R, Yoshihara A, Matsubara T, et al. Deep learning for stock prediction using numerical and textual information[C]// IEEE/ACIS, International Conference on Computer and Information Science. IEEE, 2016:1-6.

[18] Hu Z, Liu W, Bian J, et al. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction[C]//Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. ACM, 2018: 261-269.

[19] 张凯, 任维平, 张仰森,等. 基于股民评论信息的股票预测方法研究[J]. 北京信息科技大学学报(自然科学版), 2017, 32(5):67-71.

[20] 董理, 王中卿, 熊德意. 基于文本信息的股票指数预测[C]// 自然语言处理与中文计算会议. 2016.

[21] Guo K, Sun Y, Qian X. Can investor sentiment be used to predict the stock price? Dynamic analysis based on China stock market[J]. Physica A: Statistical Mechanics and its Applications, 2017, 469: 390-396.

[22] Zhang X, Shi J, Wang D, et al. Exploiting investors social network for stock prediction in China's market[J]. Journal of computational science, 2018, 28: 294-303.

[23] Zhang X, Zhang Y, Wang S, et al. Improving stock market prediction via heterogeneous information fusion[J]. Knowledge-Based Systems, 2018, 143: 236-247.

[24] Huang J, Zhang Y, Zhang J, et al. A Tensor-Based Sub-Mode Coordinate Algorithm for Stock Prediction[J]. arXiv preprint arXiv:1805.07979, 2018.

[25] Das S, Behera R K, Kumar M, et al. Real-Time Sentiment Analysis of Twitter Streaming data for Stock Prediction[J]. Procedia Computer Science, 2018, 132:956-964.

[26] Harris Z S. Distributional Structure[J]. Word, 1981, 10(2-3):146-162.

[27] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]// International Conference on Neural Information Processing

Systems. Curran Associates Inc. 2013:3111-3119.

[28] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[29] Bengio Y, Schwenk H, Senécal J, et al. Neural Probabilistic Language Models[J]. Journal of Machine Learning Research, 2003, 3(6):1137-1155.

[30] Pennington J, Socher R, Manning C. Glove: Global Vectors for Word Representation[C]// Conference on Empirical Methods in Natural Language Processing. 2014:1532-1543.

[31] Rong X. word2vec parameter learning explained[J]. arXiv preprint arXiv:1411.2738, 2014.

[32] Maas A L, Daly R E, Pham P T, et al. Learning word vectors for sentiment analysis[C]//Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1. Association for Computational Linguistics, 2011: 142-150.

[33] Yu L C, Wang J, Lai K R, et al. Refining Word Embeddings Using Intensity Scores for Sentiment Analysis[J]. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP), 2018, 26(3): 671-681.

[34] Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain.[J]. Psychological Review, 1958, 65(6):386-408.

[35] Rumelhart, David E, Hinton, et al. Learning representations by back-propagating errors[J]. 1986, 323(6088):399-421.

[36] Jarrett K, Kavukcuoglu K, Ranzato M, et al. What is the best multi-stage architecture for object recognition?[C]// IEEE, International Conference on Computer Vision. IEEE Computer Society, 2009:2146-2153.

[37] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines[C]// International Conference on International Conference on Machine Learning. Omnipress, 2010:807-814.

[38] He K, Zhang X, Ren S, et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification[J]. 2015:1026-1034.

[39] Lecun Y, Boser B, Denker J, et al. Backpropagation Applied to Handwritten Zip Code Recognition[J]. Neural Computation, 2014, 1(4):541-551.

[40] Kim Y. Convolutional neural networks for sentence classification[J]. arXiv preprint arXiv:1408.5882, 2014.

[41] Rumelhart, David E, Hinton, et al. Learning representations by back-propagating errors[J].

1986, 323(6088):399-421.

[42] Hochreiter, S, Schmidhuber, J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735-1780.

[43] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv:1406.1078, 2014.

[44]Schuster M, Paliwal K K. Bidirectional recurrent neural networks[M]. IEEE Press, 1997.

[45] Lin Z, Feng M, Santos C N D, et al. A Structured Self-attentive Sentence Embedding[C]// International Conference on Learning Representations. 2017.

[46] Yang Z, Yang D, Dyer C, et al. Hierarchical attention networks for document classification[C]// Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016: 1480-1489.

[47] Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift[J]. 2015:448-456.

[48] Srivastava R K, Greff K, Schmidhuber J. Highway networks[J]. arXiv preprint arXiv:1505.00387, 2015.

[49] Abadi M, Agarwal A, Barham P, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems[J]. 2016.

[50] Tang Y. TF. Learn: TensorFlow's high-level module for distributed machine learning[J]. arXiv preprint arXiv:1612.04251, 2016.

[51] Bahrampour S, Ramakrishnan N, Schott L, et al. Comparative Study of Deep Learning Software Frameworks[J]. Computer Science, 2016.

[52] Abadi M, Barham P, Chen J, et al. Tensorflow: a system for large-scale machine learning[C]// OSDI. 2016, 16: 265-283.

[53] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. Journal of Machine Learning Research, 2014, 15(1):1929-1958.

[54] Pascanu, Razvan, Mikolov, Tomas, Bengio, Yoshua. On the difficulty of training Recurrent Neural Networks[J]. 2012, 52(3):III-1310.

[55] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

致 谢

时光荏苒，岁月如梭，转眼间研究生生涯即将结束。在这两年半的学习中，我收获了很多，不仅学习到了最前沿的知识技术，还有一些学习方法与人生经验。感谢东北大学给我攻读硕士学位的机会，“自强不息，知行合一”的校训我将铭记于心。

感谢我的导师崔建江老师。本文是在崔老师的精心指导下完成的，在课题的研究与论文的完成过程中，崔老师总是十分耐心地与我讨论问题，并提出指导意见。在科研方面，崔老师认真负责，学术态度严谨；在教育学生方面，崔老师关心每一位学生，并通过言传身教告诉我们，做人做事都要踏踏实实，脚踏实地。

感谢实验室的上级师兄田源与刘佩宝，以及同级同学王豪、王邦新、刘硕、杨志谋等人对我研究工作的支持和帮助。在平时的学术讨论中，他们提出了很多创新性的建议与看法，为我的研究注入了新的思想。

感谢我的家人在生活中对我给予的关心与支持，正是他们多年来的理解与关心支持着我走到了今天。

另外感谢文本的数据获取来源，Tushare 金融大数据社区与雪球网。

最后向所有对本文的完成提供过帮助的人表示感谢！