学校代码：10272

学　　号：2019210933

上海财经大学

SHANGHAI UNIVERSITY OF FINANCE AND ECONOMICS

硕士学位论文

MASTER　DISSERTATION

论文题目	基于 BERT 模型的金融新闻舆情对股票收益率影响研究
作者姓名	孙明宏
院(系所)	金融学院
专　　业	金融硕士
指导教师	陈云
完成日期	2021 年 5 月 15 日

学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本人的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

论文作者签名：孙明宏　　日期：2021 年 5 月 15 日

学位论文版权使用授权书

（硕士学位论文用）

本人完全了解上海财经大学关于收集、保存、使用学位论文的规定，即：按照有关要求提交学位论文的印刷本和电子版本。上海财经大学有权保留并向国家有关部门或机构送交本论文的复印件和扫描件，允许论文被查阅和借阅。本人授权上海财经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和传播，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

论文作者签名：孙明宏　　导师签名：

日期：2021 年 5 月 15 日　　日期：2021 年 5 月 15 日

摘要

在金融市场中，金融新闻承担着传播和评价有关公司管理、高管言论、企业经营情况的任务，在市场中扮演着重要的作用，市场中的交易者在接收到金融新闻所传递的信息之后，会根据其自身的判断和观念形成对事件的认知，并依据这一认知采取相应的行动，因此，研究新闻情绪，特别是研究网络金融新闻舆情对于社会具有重大意义。

现有的研究往往通过构建情绪词典或者依据词频提取语言特征，并采用机器学习模型提取情绪的方式来实现新闻舆情的识别与抽取，虽然这类方法的可解释性相对较强，但由于这类方法基于词袋模型的假设，忽略了词法和语法对于语句内容和情绪表达的影响，致使该类方法在识别情绪准确率上存在较大误差。而在深度学习自然语言处理模型中词的分布式表征方法和卷积、循环、Transformer 神经网络的运用能够更好抽取和利用文本中的语义信息，在大量公开数据集上的研究表明，基于深度学习的情绪识别模型在准确率上要强于传统方法。BERT 模型是基于预训练的神经网络模型，这一模型通过使用大量外部文本进行自监督预训练的方式为参数学习到良好的先验状态，从而能够在下游监督学习任务上产生较好的表现，而 Gururangan（2020）则提出使用领域内外部文本在自监督预训练后再次进行预训练的方法，能够进一步提升 BERT 模型处理特定领域文本的能力。

本文分析对比了各类模型在金融新闻舆情识别与抽取准确性上的差异并通过实验证明了在经过金融文本适应性预训练后，BERT 具有较好的舆情识别能力；通过运用该模型在涉及个股的新闻进行舆情识别后构建金融新闻舆情指标，在 2017 年到 2020 年上半年的历史数据上回测并分析信息系数、分组回测表现并与中证 500 指数进行对比，发现基于 BERT 模型的金融新闻舆情指标对股票市场收益率具有较好的预测能力。

为了研究对比各类模型金融新闻舆情抽取与识别的性能，本文通过人工标注和外部混合的方式获取一万三千条新闻舆情样本，将每条样本标注为积极、中性或消极中的一类，并分别使用基于传统机器学习的朴素贝叶斯模型、TF-IDF 特征提取与逻辑回归模型和基于深度学习的 TextCNN、TextRCNN、AttentionLSTM

等经典深度学习模型与基于预训练的 BERT 模型以及在金融文本上进行适应性预训练的 BERT 模型在上述样本上进行训练与测试，并通过准确率和 F1 两个评价指标评价模型的表现；通过对比基于机器学习模型、经典深度学习模型、未进行适应性预训练的 BERT 模型和使用外部金融文本进行适应性预训练的模型在上述样本上的表现，本文得到加入外部金融文本进行领域内适应性预训练的 BERT 模型在识别情绪的 F1 指标和准确率指标上表现超过其他模型的结论。

为了计算个股在每日的金融新闻舆情指标，本文使用加入外部金融文本进行领域内适应性预训练的 BERT 模型对三百余万条涉及个股的金融新闻进行舆情识别，得到每条金融新闻在正面、中性、负面情绪上的概率分布，计算金融新闻情绪指标，并平均当日涉及个股的所有新闻情绪指标并在时间序列上取十四个自然日的简单平均，得到最终金融新闻舆情指标。

为了研究基于 BERT 的金融新闻舆情指标对股票市场收益率的影响，本文采用信息系数及相关指标研究金融新闻舆情指标与其下一期收益率之间的相关性以及相关性的稳定程度；使用分组回测的方式，每期将新闻舆情指标取值不为空的股票分为五组分别持有并在下一期进行调仓，验证该指标对于个股收益率预测的单调性；使用金融新闻舆情指标对下一期收益率进行回归并验证该指标的预测能力，除此之外，还利用金融新闻舆情指标构建策略并在考虑冲击成本和交易成本的基础上进行回测；实证研究表明金融新闻舆情指标与下一期收益率之间相关性较强，而相对极端的金融新闻舆情能够对个股收益率具有较好的区分效果，回测结果显示，通过金融新闻舆情指标构建的策略显著优于持有中证 500 指数的基准，由此得出金融新闻舆情指标对个股下一期收益率具有较好的预测能力的结论。

关键词：金融新闻舆情　BERT 模型　预训练　股票市场

Abstract

Traders in the market will form their own perceptions of events based on their own judgments and perceptions after receiving the information conveyed by financial news, and take corresponding actions based on this perception. Therefore, it is of great significance to society to study news sentiment, especially to study online financial news opinion.

Although these methods are relatively interpretable, they are based on the assumption of bag-of-words model and ignore the influence of lexical and grammatical aspects on the content and emotion expression of statements, resulting in a large error in the accuracy of emotion recognition. However, because these methods are based on the assumption of bag-of-words model, they ignore the influence of lexis and grammar on the content and emotion expressions, resulting in a large error in the accuracy of emotion recognition. The BERT model is based on a pre-trained neural network model, and this model uses a large amount of external text for self-training. model learns a good prior state for the parameters by using a large amount of external text for self-supervised pre-training, which can produce better performance on downstream supervised learning tasks, while Gururangan (2020) proposes the use of external text in the domain for pre-training again after self-supervised pre-training, which can further enhance the BERT model's ability to process domain-specific text capability.

This paper analyzes and compares the differences of various models in the accuracy of financial news opinion identification and extraction and proves through experiments that BERT has better opinion identification ability after adaptive pre-training of financial texts; by using the model to construct financial news opinion indicators after opinion identification of news involving individual stocks, back-testing and analyzing information coefficients, grouping back-testing performance on historical data from 2017 to the first half of 2020 and comparing with CSI 500 index, it is found that financial news opinion indicators based on BERT model have better prediction ability for stock market returns.

To investigate the performance of various models for financial news opinion prediction and recognition, this paper obtains 13,000 news opinion samples by manual labeling and external mixing, labeling each sample as one of: positive, neutral, or negative, and using traditional machine learning-based plain Bayesian model, TF-IDF feature extraction and logistic regression model, and deep learning-based TextCNN, TextRCNN and AttentionLSTM The performance of the models is evaluated by two evaluation metrics: accuracy and F1. By comparing the performance of machine learning-based models, classical deep learning models, BERT models without adaptation pre-training and models with adaptation pre-training using external financial texts on the outside samples, this paper concludes that BERT models with in-domain adaptation pre-training by adding external financial texts outperform other models in the F1 metrics and accuracy metrics of sentiment recognition.

In order to calculate the daily financial news sentiment indicators of individual stocks, this paper uses the BERT model with external financial texts for in-domain adaptive pre-training to identify the sentiment of more than three million financial news involving individual stocks, obtains the probability distribution of positive, neutral and negative sentiment for each financial news, calculates the financial news sentiment indicators, and averages all the news sentiment indicators involving individual stocks on the same day and takes them in time The final financial news sentiment indicators are obtained by taking a simple average of fourteen natural days in the sequence.

In order to study the impact of BERT-based financial news sentiment indicators on stock market returns, this paper uses information coefficients and information ratios to investigate the correlation between financial news sentiment indicators and their returns in the next period and the stability of the correlation; using group backtesting, stocks with non-null values of news sentiment indicators in each period are divided into five groups and held separately in the next period to verify the indicator's In addition, we also use financial news opinion indicators to construct strategies and conduct backtesting based on shock costs and transaction costs. The backtest results

show that the strategy constructed by financial news opinion indicators significantly outperforms the benchmark holding the CSI 500 index, which leads to the conclusion that financial news opinion indicators have good predictive power for next period returns of individual stocks.

Key Words: financial news sentiment BERT Pretraining stock market

目 录

第一章　引言

第一节　研究背景

新闻媒体在人们的社会生活中扮演着信息中介的角色，它作为信息的发送端，在收集到社会事件后进行加工和处理形成新闻，并把新闻通过其自身的渠道进行报道，成为社会信息的重要收集者、加工者和传播者；社会公众在接收到经过新闻媒体加工报导的社会事件后，会根据自身的判断和观念产生对社会事件的认知，并在一定程度上影响其判断和行动。在金融市场中，金融新闻是金融市场的重要传声筒，承担着传播有关公司管理、公司高管言论、企业经营状况等信息并加以处理的任务，通过对于金融新闻舆情的准确识别和研究新闻情绪与股票收益的关系，可以使得投资者和监管层更加确切的理解市场情绪的波动，并在一定程度上能够及时采取防范风险的措施，对理解股票市场波动和防范风险有重要意义。

在传统的金融学理论中，研究者往往会做出市场上的投资者都是理性的，投资者通过权衡风险和收益来做出理性的投资决策，研究者们从该假设出发推导出经典的金融学理论，但是在实际市场中，投资者并非完全理性，并且对未来的收益的预期存在系统性的偏差，这种偏差被称作投资者情绪，随着行为金融学的发展，股票收益与投资者情绪的关系就逐渐成为国内外研究者研究的热点内容。当前关于情绪指标的构建方法较多，较为直接的情绪指标包括消费者信心指数，美国个体投资者协会指数(Fisher &Statman,2000 4)投资者智能指数(Siegel，1992 5）等等，除了直接情绪指标，有些学者提出使用间接情绪指标来衡量投资者情绪，包括封闭式基金折价率（Brown，1999 6)、IPO 首日收益（Ljungqvist 等，2006 7)、成交量（Brown&Cliff，2004 8)、市场流动性水平（Baker&Stein，2004 9）等，这些基于直接情绪指标和间接情绪指标得出的结论往往是：投资者情绪对股票的收益具有正向的影响，但是这些指标的研究没有充分利用更加直观、距离投资者更近的公共媒体或个人发布的新闻或者相关评论文本信息来提取情绪。

随着研究的不断深入，有学者发现除了简单的利用代理指标或者直接指标

来计算投资者情绪之外，金融相关的文本中也包含了许多关于投资者情绪的信息，近年来，越来越多的学者通过将文本挖掘技术引入到市场与投资者情绪研究的领域上来，Antweiler 等(2004)将来自雅虎财经的帖子按照贝叶斯分类法分为看多、持平以 及看空三类，以此构建反映投资者情绪的投资者情绪指数，李岩和金德环通过使用朴素贝叶斯模型从对股吧评论进行文本分析，从个股层面出发研究投资者情绪，通过实证研究得出了投资者情绪对股票收益具有较为显著的影响这一结论（2019），易洪波等使用基于情绪词典的情绪判别方法构建出投资者情绪指标。除了财经和股票论坛的发帖信息，公共信息的传播也会在一定程度上影响投资者的交易行为，新闻报道就是这类公共信息的代表（Wei 等，2016），在金融市场中，新闻媒体作为传播重要信息的媒介，会向多类型的市场参与者传达重要的信息，这些信息一般会涵盖国家和政府政策的发布实施、股市行情的评价、重要人物讲话、上市公司重大决策等，对于新闻媒体在对公司的报道中传递出对其经营现状、盈利预期、未来发证等方面的积极或消极的内容，称之为“新闻舆情”，于琴等通过构建中文情感极性词典的方式划分情感强度和情感极性，得到了积极的新闻舆情能够有效预测下一周股票收益率的上涨的结论。

上述研究对于情感抽取和判别的方法往往集中在使用情感词典等经典统计方法和朴素贝叶斯模型等传统机器学习领域，这些方法往往基于词袋模型假设，即认为词和词之间不存在联动关系，将每个词视作独立的个体，通过词频统计方式得到对于情感的判别，这类方法在处理复杂文本和语义反转等情况下效果一般，准确率不高，而另一方面，随着深度学习在自然语言处理中的情感抽取、分析和判别领域的广泛利用，一些深度学习模型在情感抽取和判别的准确率上已经被证明显著高于基于词频统计的方法，在卷积神经网络和循环神经网络被应用于文本情绪识别之后，谷歌在 2019 年提出的 BERT 模型通过在大量外部文本进行预训练并在目标数据集上微调的方式，使得基于 BERT 的各种文本挖掘任务的表现有了普遍 5%到 10%的提升，为了更加充分的挖掘文本信息和提高情绪抽取识别的表现，本文通过对比近年来业界和学术界常用的机器学习和深度学习模型并证明了 BERT 模型在情绪识别准确性上的优势，然后使用基于 BERT 的文本情绪抽取和判别方法量化金融新闻舆情并构建金融新闻舆情指标，来分析其与股票收益率的关系。

第二节　研究意义

本文具有一定的理论意义，其一，本文从新闻媒体的视角来关注投资者情绪，从更为广义的视角分析其对于我国股票市场收益率的影响；其二，本文在情感抽取和判别数据集上验证并使用了更为先进的 BERT 模型，并使用大量外部金融文本对 BERT 模型进行进一步预训练以调整其内部语言主题分布，能够更加准确的识别公共媒体情绪；其三，本文通过构建金融新闻舆情指标并进行实证，详细分析了新闻舆情对股票收益的影响。

本文还具有一定的实践意义：从“互联网+”政策实施以来，网络媒体在新闻传播的过程中发挥的作用越来越大，通过正确识别新闻报道中的舆情信息，并在适当理解其与股票市场中股票收益之间的关系，在一定程度上有利于投资决策，另外，引入新闻舆情信息能够为政府监管部门指定相应的政策提供参考，从而维护市场的秩序和稳定。

第三节　论文创新点

相比起前人使用文本挖掘构建情绪指标和对该情绪指标与股市相关性的研究而言，本文的贡献主要体现在以下三点：一是相对于研究传统投资者情绪和间接指标的行为金融学研究方法，本文主要分析和利用以财经新闻为代表的公共信息所包含的舆情信息是否会影响股票价格，在一定程度上丰富了利用文本挖掘与分析的内容；二是相比起传统基于词典和机器学习的统计方法，本文采用深度学习自然语言处理中表现良好的 BERT 模型，在通过金融领域文本对 BERT 参数进行进一步预训练并调整 BERT 模型内部的语义分布，来对情绪信息进行识别，在一定程度上提升了新闻舆情抽取和识别任务的准确性和可信度。三是本文使用 BERT 情绪抽取与判别模型在 2018 年到 2020 年 6 月 30 日的金融新闻数据上进行预测并构造情绪因子，并研究了金融新闻舆情指标对股票收益的预测能力。

第四节　结构安排

本文结构安排如下：第二章将对市场情绪和新闻舆情的研究进行综述、对利用深度学习进行情绪抽取领域进行综述，并说明舆情研究中存在的准确性问题，

第三章将介绍本文研究设计内容，本文首先会在本章中介绍情感抽取和识别模型，包括基于机器学习和深度学习的情绪识别与文本分类模型，并介绍将使用金融文本对 BERT 模型进行语义分布调整的方法，然后会介绍用于情感分类的数据集并使用前文介绍的模型在该数据集上进行验证，选择出对于情绪识别准确性最高的模型来供后续使用，第四章包括通过情绪识别模型对 2017-2020 年三百多万条与个股相关的新闻文本进行识别并计算金融新闻舆情指标，并对金融新闻舆情指标在选股能力和预测能力上的表现进行实证研究，第五章包括本文的主要结论，对投资者和政策的相关建议和进一步研究的方向展望。

第二章　文献综述

伴随着近年来网络媒体的不断发展，社交媒体和网络平台媒体等传播媒介正逐渐成为研究者们关注的热点，通过对市场信息即时快捷的报道，这些媒体向新闻的接收者传递了对市场信息的评价和对未来走势的乐观或者悲观的估计，学者们也逐渐通过文本挖掘技术，通过将文本分析方法应用到新闻媒体的文本数据上，研究投资者情绪和新闻舆情对市场的影响。本文的主题是使用深度学习中自然语言处理模型来对新闻舆情进行抽取和识别，并使用模型所计算的金融新闻舆情指标来研究新闻舆情与股票收益与股票收益的关联性，相关的文献主要包括以下三个方面：一是市场情绪的衡量方法，二是新闻舆情对股票市场收益率的影响，三是用于识别和抽取新闻中所包含情绪的文本挖掘技术。

第一节　市场情绪的衡量方法

从行为金融学的角度来看，参与市场的投资者的理性程度是有限的，人们在有限理性的情况下根据市场信息和反馈做出决策，Lee et al.（1991）提出使用封闭式基金折价率衡量投资者情绪，Baker & Stein（2004）首次提出去趋势的对数换手率来代表投资者情绪，Ibbotson et al（1994）则采用了 IPO 公司数量和 IPO 公司上市首日平均收益指标来对投资者情绪进行衡量，Baker and Wurgler（2006）使用了前人提出的六个指标通过两步法计算出投资者情绪，Huang et al（2015）指出了利用主成分分析方法存在的缺陷并使用偏最小二乘法来估计投资者情绪，这些计算投资者情绪的方法往往是利用一些经济指标和数据作为投资者情绪的衡量；除了利用一些经济指标作为投资者情绪的代理之外，利用文本挖掘衡量投资者情绪的研究随着文本挖掘技术的成熟和发展也有所发展，主要包含三个方面，即利用 Twitter、Facebook 和微博等社交媒体为主的社交平台文本挖掘和情绪识别，利用股吧论坛等金融论坛为主的发帖者情绪识别，以金融新闻为主的新闻文本分析；Zhang（2011）等人通过收集六个月的 Twitter 信息并分析其情绪倾向，发现情绪化的 Twitter 与道琼斯、纳斯达克和标普 500 指数显著负相关，黄润鹏等（2015）利用新浪微博平台的接口分析微博情绪与上证综指的相关性，发现微博情绪指标的加入可以有效提升模型的准确

率，郑瑶（2016）认为投资者情绪具有异质性，利用东方财富网股吧发帖文本，使用情绪关键词统计的方法衡量发帖的投资者整体情绪强度和情绪倾向并对投资者情绪异质性程度进行衡量，李岩和金德环（2017）通过爬虫抓取 2010-2016 年东方财富网股吧的评论数据并使用朴素贝叶斯方法训练情绪判别模型，计算得到市场情绪，张凯等（2017）则直接使用股吧股民评论进行文本挖掘并将文本信息用于股价预测。

上述研究主要通过两个方法来研究投资者情绪，即使用投资者情绪的代理变量和直接对传递投资者情绪的股吧等平台进行文本挖掘，这两者的研究都存在一定的问题，前者通过代理变量表示投资者情绪存在的问题主要是：使用一个或者多个代理变量只能从某些特定的方面来对投资者情绪进行衡量，而不能完全涵盖投资者的实际情绪；后者通过文本挖掘和情绪识别的方法能够更加直观地获取投资者情绪，但其数据源具有一定的有偏性，即在股吧发帖或者在相关社交媒体上发声的人可能只是投资者的子集，并非是投资者整体的无偏抽样，从而利用这类数据得到的投资者情绪也并不能完全涵盖全市场投资者情绪；而新闻由于其受众的广泛性，对新闻舆情的研究则更具有普遍意义。

第二节　新闻效应和新闻舆情对股市收益率影响的研究

对股票市场中的新闻效应和新闻舆情作用的研究如下：Tetlock（2007）运用文本分析的方法从华尔街日报的金融新闻中构建媒体悲观指标，Tetlock et al.（2008）则利用提出利用哈佛词库对新闻的情绪指数进行打分，Hillert et al.（2014）使用了 1989 到 2010 年报刊的 220 万篇新闻，使用文本挖掘技术研究了关注度与动量、反转效应之间的关系，研究了新闻异质性和股票短期收益之间的关系，从政策扶持、兼并收购、再融资、盈利能力和违规处罚五个方面区分了互联网新闻并分析在互联网财经新闻异质性下的股票市场价格运作机制，Rognone 等（2020）研究了新闻舆情与传统货币和比特币收益率、波动率和交易量的关系，Shi 和 Ho（2020）通过使用道琼斯新闻分析数据库，研究新闻情绪对道琼斯综合指数成分股波动状态变化的影响，结果发现宏观经济新闻和公司特定新闻的情绪会显著影响股票收益率的波动性；除了国外学者对于新闻情绪的研究，国内对于新闻情绪和股票市场的研究起步相对较晚，但也有较为丰富的内容，杨继东（2007）在回顾了研究新闻媒体与股票价格相关的文献之后，归纳了

新闻媒体报道对股价的影响路径，他认为新闻媒体会通过影响投资者的决策，这些决策最终传导到股票市场并通过股票收益率的变动表现出来，赵丽丽、赵茜倩、杨娟等（2012）利用文本挖掘相关算法和 SVM 的机器学习模型量化财经新闻文本，在考虑沪深两市以及上市公司规模不同的因素下，研究了互联网财经新闻对股票市场的影响强度和持续时间，得到了财经新闻对深市影响更大并且公司规模程度与其受新闻影响程度成负相关的结论，。孟雪井、杨亚飞，赵新泉（2016）通过挖掘新闻关键词研究了关键词与股票价格上涨下跌之间的关系并根据关键词计算的情绪指标构建了模拟交易的策略，除了研究新闻情绪之外，也有学者研究了新闻报道和关注程度对股价的影响，杨洁、詹文杰等（2016）的研究探索了新闻报道数量和股价非同步性之间的关系，他们的研究结果表明新闻特别是财经新闻报道的数量对股价波动的非同步性的影响是“U”型的，周冬华，魏灵慧（2017）则研究了非金融类上市公司的媒体报道数据，研究表明非金融类上市公司的财经新闻媒体报道数量和股价同步性上存在显著的负相关关系。

尽管对于新闻效应和新闻舆情对股票市场的影响有许多研究，但这些研究往往基于情绪词典或者是经典的机器学习方法，随着深度学习的发展，大量研究和实践表明这类传统方法在识别情绪的准确性上存在较大误差，而基于深度学习的方法的识别准确性要显著高于传统方法。

第三节　新闻舆情识别文本挖掘技术

文本数据往往是非结构化的，相对于能够通过二维的样本-特征的表格式结构来表达信息的结构化数据，从非结构化数据中提取情绪信息的关建在于如何能够把文本中复杂的字词、词法、语法和句法信息在尽可能多地保留的前提下准确的抽取出来，在应用于自然语言处理的深度学习模型得到广泛关注之前，主要的研究者往往使用情绪词典或者 Bag of Words 模型，前者通过预先准备好的情绪词典统计语句中出现积极词汇和消极词汇的数量并依据词汇对应的情感分数进行打分，Tetlock（2008）通过使用 Harvad-IV-4 词库构建情绪词-分数词典并对新闻信息进行打分将其分为积极和消极，Loughran 和 McDonald（2016）也通过情感词典来捕捉财务披露和新闻文本中包含的情绪；后者通过计算词频将文本内容简化为一个以文本为行，文字频数为列的稀疏矩阵的方式来简化文本信息，然后机器学习方法对抽取后的文本特征矩阵和标签向量学习联合分布，这种

方法可以解决广义上的文本分类问题。

随着深度学习的发展，将深度学习应用在情绪识别和文本分类领域的研究逐渐成为主流，Kim（2014）首次将卷积神经网络引入情绪识别领域，他使用一维卷积神经网络自动学习并抽取 n 元语法特征并进行分类，开创了使用卷积神经网络处理文本研究的先河，Lai（2015）则提出 TextRCNN，使用 Bi-LSTM 模型和最大池化技术来提取全局上下文语义特征，并利用双向循环网络一定程度上缓解了单向 LSTM 的遗忘问题，Zhou 等人（2016）在处理知识图谱关系分类的问题中提出了基于注意力机制的 LSTM 模型，后来该模型被用于情感识别领域并有了良好的表现。Devlin 等人（2019）提出使用迁移学习的方式，利用 Vaswani（2017）提出的 Transformer 模型中的编码器部分作为神经网络模型的主要结构，通过该编码器双向提取文本表征，并使用 MLM 和 SOP 预训练任务在大量外部文本上进行训练并在具体数据集上进行微调的预训练-微调模式使得其在大多数 NLP 标准数据集上的表现都产生了质的飞跃。

综上所述，对投资者情绪和市场舆情的研究随着技术的发展不断变化，从间接指标到直接挖掘股吧论坛等文本信息，研究者对于市场情绪的识别越来越直接和深入，但是识别投资者情绪和新闻舆情的手段却停留在情绪词典和传统机器学习方法上，然而情绪词典方法存在较大的主观性且人为总结情感词典工作量巨大，传统机器学习模型的识别准确性欠佳，在实践上被证明准确程度更高的深度学习方法却在使用较少，为了对这一问题进行补充完善，并更加准确的识别情绪，本文将测试传统机器学习模型和经典深度学习模型在新闻舆情识别任务上的表现，并将其与在金融文本上进行进一步预训练的 BERT 模型做对比，实验证明本文使用的 BERT 模型在金融文本上进行适应性预训练后，在识别情绪准确性上超越了传统机器学习方法，并使用训练好的 BERT 模型对 2017-2020 年涉及个股的 300 多万条新闻信息进行情绪识别并构造情绪指标，通过 Qian 提出的因子分析框架来分析基于金融新闻舆情指标的选股效果和对于股票市场中个股收益率的预测能力，实证结果基于 BERT 模型构建的金融新闻舆情指标对于股票市场收益率具有较强的预测能力。

第三章　金融新闻舆情模型研究

第一节　情绪识别模型

由于基于词典的方法需要较多的人力物力，且对于每个词的情绪评分主观性较强难以量化，本文接下来主要介绍和对比基于词袋模型和机器学习方法的情感抽取和识别模型以及深度学习自然语言处理领域的情感识别模型。

一、基于传统机器学习方法的情绪识别模型

基于机器学习方法的情感识别模型往往被归类为统计自然语言处理中经典的 Bag of Words 模型，该模型往往运用于信息检索领域，它假设对于一篇文档或者一个句子，忽略其词序、语法等相关信息，而仅仅将文档或者句子当作是词或字的集合，文本中每个词汇相互独立。基于 Bag of Words 模型往往需要对全文档进行统计，在获取到文档的词表和词频之后，通过词频抽取文档特征进行分类。

朴素贝叶斯算法以贝叶斯原理为基础，使用统计的方法对文本进行分类，相比起贝叶斯算法，朴素贝叶斯的特点在于其假设的简化，即假定在给定标签时各个特征之间相互条件独立，各个特征在决策结果中的比重相同，这一假设在一定程度上降低了算法的分类结果，但由于其计算高效，极大简化了贝叶斯方法的复杂性并在实践中得到广泛运用。在文本情绪识别领域，朴素贝叶斯算法通过预先给定的训练集，在统计词频作为文本特征之后，假设输入的词之间相互独立，学习特征与标签的联合分布，在预测过程中，基于学习到的模型，计算得到使得后验概率最大的标签值作为预测值。

简单统计词频作为文本特征的方式有时会受到高频无意义词的影响，为了解决这一问题，TF-IDF 方法通过评估一个词或字对于一份语料库或文档集合的相对重要程度来抽取模型特征，通过计算词频（TF）与逆文档频率（IDF）相乘的方式，使得字词的重要性随着其在单文档中出现的频率增加而增加，随着其在语料库或文档集合中出现的频率的增加而下降，具体的，对于在单文档中出现的词t_i，词频（TF）可以表示为：

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{3.1}$$

其中$n_{i,j}$为该词在文档d_j中出现的次数，分母部分是该文档中所有词出现的总次数。类似的，逆文档频率可以表示为：

$$IDF_i = \lg\left(\frac{|D|}{|\{j:T_i \in d_j\}|}\right) \tag{3.2}$$

其中$|D|$为总文档数，分母为出现过词 i 的文档数，最后，TF-IDF 抽取的文档特征则可以表示为$TFIDF_{i,j} = TF_{i,j} \times IDF_i$，最后通过逻辑回归或者支持向量机的有监督分类模型对被抽取出文本特征的句子进行文本分类和情绪识别。

二、基于非预训练深度学习的情绪识别模型

TextCNN 模型将卷积神经网络引入文本和情感识别领域，卷积神经网络由于其稀疏连接、参数共享和平移不变性表示，往往在深度学习中被用于抽取局部特征，TextCNN 则是使用一维卷积的方式，通过宽度固定的卷积核在经过词向量嵌入之后的文本特征上进行卷积并提取局部 n 元词法特征，然后使用最大池化的方法抽取出局部 n 元词法特征中最重要的部分，使用激活函数增加非线性之后，将抽取出的特征送入全连接层进行分类学习，相比起其他深度学习模型，TextCNN 的最大优势在于由于网络结构简单带来的参数数目少、计算量远小于其他深度学习模型，在训练速度和收敛速度上具有很大的优势。

TextRCNN 同时将循环神经网络和卷积神经网络中的最大池化组件引入文本分类和情感识别领域，该模型使用 LSTM 神经网络模型来处理输入的词序列，LSTM 通过其内部的隐藏层储存了整个序列的内部信息，从而能够更好的获得句子上下文信息，但是 LSTM 模型是有偏的模型，存在一定的遗忘效果，所以 TextRCNN 使用双向 LSTM 模型，分别提取文本的顺序和逆序信息，并将提取出的信息使用最大池化来抽取出结合了上下文的全局语义特征，并将全局语义特征输入全连接层进行分类，TextRCNN 在一定程度上缓解了 TextCNN 中无法提取全局语义的问题，并通过双向循环神经网络模型得到各个位置的词的表征，丰富了神经网络抽取语义的内容，并在公开数据集上的验证结果超越了 TextCNN。

AttentionLSTM 模型在使用双向 LSTM（Bi-LSTM）的基础之上加入注意力机制，注意力机制的引入目的是为了解决传统循环神经网络的遗忘问题，该机制

通过参数矩阵学习模型在输入文本序列上每个词所在位置应当被赋予的权重，并将经典的最大池化或平均池化归纳信息的方法变为对每个文本序列上 Bi-LSTM 编码后词向量加权平均，在一定程度上减轻了最大池化丢失信息的问题和平均池化中每个 Bi-LSTM 编码后词向量对文本特征向量贡献相等的缺陷，使得模型能够选取更重要位置的词来提升情绪分类表现，并且，注意力机制的引入使得使用者能够通过观察注意力矩阵得到文本中对于输出贡献较大的词，从而对模型的输出内容做出适当的解释。

三、BERT 模型

BERT 模型采用 Vaswani（2017）提出的 Transformer 编码器作为神经网络模型的基础结构，相比起之前提到的模型通过循环神经网络和卷积神经网络抽取语义信息的方法，Transformer 模型通过采用多头自注意力机制，通过语义权重计算的方法实现了相对高效的文本信息抽取，并基本上解决了卷积神经网络只能捕捉局部 n 元语法信息的缺陷和循环神经网络的信息有偏和处理长文本时存在的遗忘问题，而 BERT 模型在本质上是多层 Transformer 编码层模型的堆叠。通过多层神经网络结构逐层深化对文本的理解和抽象能力，并最终生成富含文本内容语义信息的张量。这种对文本的理解和深化是通过预训练完成的，预训练是使用模型和大量无标签文本在预训练任务上进行训练，从而使得被训练的模型能够调整参数并在一定程度上学习到语义相关信息，在 BERT 模型中，主要采用了两个预训练任务，包括 MLM（Masked Language Model）任务和 SOP（Sentence Order Prediction）任务，BERT 会首先对输入句子进行预处理，在句首加入特殊符号[CLS]，在句尾加入特殊符号[SEP]，MLM 任务是指在预训练过程中，对于输入模型的句子中的每个字或词，以 15%的概率进行名义上替换，对于被替换的词，以 80%的概率使用特殊符号[MASK]来进行替换，以 10%的概率使用词典中其他词进行替换，以 10%的概率不替换，并将替换后的句子输入模型，模型的标签则是原始未经替换过的句子，这一预训练任务的目的是使模型能够充分理解上下文并根据上下文输出合适的填补词；SOP 任务是指对于每个句子，以 50%的概率进行替换，被替换的句子使用其他句子的后半句代替原本句子的后半句，将被替换句子数据输入模型之后，使用模型输出的[CLS]位置的向量输入到全连接层判别该句是否被替换过，通过这一预训练任务，模型则能够捕获全局语义信息并

识别语义信息的断层。经过预训练的任务往往包含了关于语言本身的语法、词法、句法信息，也为后续的微调选取到合适的初始参数值。在后续文本分类任务中，使用者通过在原有模型的基础上在新样本上使用较低的学习率进行有监督训练，往往能够获得较好的模型表现。具体的：

对于输入的句子进行嵌入可以得到词向量矩阵$\mathrm{X} = [x_1, x_2, \ldots, x_n], \mathrm{X} \in R^{n \times d}$，将词向量矩阵按顺序输入 12 层堆叠的 Transformer 编码层模型进行上下文编码，令$\mathrm{X} = H^{(0)}$得到隐藏层的输出如下：

$$H^{(i)} = TransformerEncoder\left(H^{(i-1)}\right) \quad \mathrm{i} = 0,1,2,\ldots,12 \tag{3.3}$$

其中$H^{(i)}$为第 i 层 Transformer 编码层模型的输出，$H^{(i)} \in R^{n \times d}$，在分类任务和情绪识别任务中，取出标签中特殊符号“[CLS]”所在位置的向量作为全局语义向量

$$h_{[cls]} = H^{(12)}{}_{t=0} \tag{3.4}$$

最后将$h_{[cls]}$输入全连接层输出在每个标签上的概率分布，在预训练任务中，则直接将$H^{(12)}$输入全连接层计算在句子中每个位置输出的概率分布并使用交叉熵作为损失函数，使用梯度下降法更新参数。

第二节　BERT 模型在金融文本上的适应性预训练

Gururangan（2020）提出，目前使用多种来源的预训练语言模型的预训练语料来自多个数据源，尽管这种基于多个领域数据源进行训练的语言模型在许多大量通用领域的数据集上表现不错，但在涉及到需要理解专业知识和专业领域内容的时候却还有提升的空间，本文为了更好利用金融领域的富文本特性，并提高模型在情绪抽取与识别任务上的表现，文本将使用网络上开源的无标签金融新闻数据集进行自监督学习并进行预训练，使用 MLM 任务构造数据集的方式如下：

对于输入句子$\mathrm{S} = [w_1, w_2, \ldots, w_{n-1}, w_n]$，在句首加入“[CLS]”的特殊符号标识句子开头，在句尾加入“[SEP]”的符号标识句子结尾，然后使用内置的预先设定好的词典对句子进行切分如下

表 3.1　词典切分结果

原始输入	切分后输出
渔光互补：从“吃鱼难”到“吃好鱼	[CLS] 渔 光 互 补 ： 从 [UNK] 吃 鱼 难 [UNK] 到 [UNK] 吃 好 鱼 [UNK] [SEP]
'贵州茅台''国酒'商标注册被否 袁仁国称我不清楚	[CLS] ' 贵 州 茅 台 ' ' 国 酒 ' 商 标 注 册 被 否 袁 仁 国 称 我 不 清 楚 [SEP]
光大证券董事长薛峰新年致辞：市场和行业见证光大新崛起	[CLS] 光 大 证 券 董 事 长 薛 峰 新 年 致 辞 ： 市 场 和 行 业 见 证 光 大 新 崛 起 [SEP]
恒天海龙控股股东兴乐集团与中弘卓业纠纷未了 双双获深交所发函关注	[CLS] 恒 天 海 龙 控 股 股 东 兴 乐 集 团 与 中 弘 卓 业 纠 纷 未 了 双 双 获 深 交 所 发 函 关 注 [SEP]
东方日升董事长林海峰新年贺辞：照亮世界点亮心灵	[CLS] 东 方 日 升 董 事 长 林 海 峰 新 年 贺 辞 ： 照 亮 世 界 点 亮 心 灵 [SEP]

然后根据切分后的句子进行字词-ID 的映射，将新闻文本通过哈希映射表示成文本 ID 的列表，具体例子如下：

表 3.2　文本 ID 映射后结果

原始输入	映射后输出
渔光互补：从“吃鱼难”到“吃好鱼	[101 3934 1045 757 6133 8038 794 100 1391 7824 7410 100 1168 100 1391 1962 7824 100 102]
'贵州茅台''国酒'商标注册被否 袁仁国称我不清楚	[101 112 6586 2336 5747 1378 112 112 1744 6983 112 1555 3403 3800 1085 6158 1415 6145 785 1744 4917 2769 679 3926 3504 102]
光大证券董事长薛峰新年致辞：市场和行业见证光大新崛起	[101 1045 1920 6395 1171 5869 752 7270 5955 2292 3173 2399 5636 6791 8038 2356 1767 1469 6121 689 6224 6395 1045 1920 3173 2307 6629 102]
恒天海龙控股股东兴乐集团与中弘卓业纠纷未了 双双获深交所发函关注	[101 2608 1921 3862 7987 2971 5500 5500 691 1069 727 7415 1730 680 704 2473 1294 689 5272 5290 3313 749 13

	52 1352 5815 3918 769 2792 1355 114 1 1068 3800 102]
东方日升董事长林海峰新年贺辞: 照亮世界 点亮心灵	[101 691 3175 3189 1285 5869 752 727 0 3360 3862 2292 3173 2399 6590 679 1 8038 4212 778 686 4518 4157 778 2 552 4130 102]

然后遍历该列表，从第一个非特殊符号标识的位置开始，对每个位置按照 15%的概率进行替换，对于被标识要替换的文本，以 70%概率使用特殊符号[MASK]在模型字典中对应的 ID 进行替换，以 15%的概率使用字典中其他字所对应的 ID 进行替换，以 15%的概率不换，并使用另外一个列表记录哪些位置的字或词被替换过，被替换过的位置的词在该记录列表中标识为 0，否则标识为 1，以上表中原始输入的第三句为例：

原句：光大证券董事长薛峰新年致辞：市场和行业见证光大新崛起

替换后句子：光大证券董事长薛峰新年致辞：[MASK][MASK]和行业见证光大新崛起

替换后句子 ID 序列：[101 1045 1920 6395 1171 5869 752 7270 5955 2292 3173 2399 5636 6791 8038 **100 100** 1469 6121 689 6224 6395 1045 1920 3173 2307 6629 102]

训练标签：[-100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 **2356 1767** -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100]

记录列表：[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 **0 0** 1 1 1 1 1 1 1 1 1 1 1]

最后构造出的数据集是以经过 mask 操作的句子 ID 列表和 MASK 位置记录列表作为输入，原本未经替换过的句子 ID 列表作为输出的数据集。

令原句 ID 序列为Y_{ori}，替换后 ID 序列为$X_{replaced}$，MASK 位置的记录列表为 M，则适应性预训练步骤可以表示为

$$Y_{pred} = BERT(X_{replaced}, \mathrm{M}) \tag{3.5}$$

$$\mathrm{loss} = \sum_{t=0}^{T} Y_{ori,t} \log(Y_{pred,t}) + (1 - Y_{ori,t}) \log(1 - Y_{pred,t}) \tag{3.6}$$

在训练过程中，使用梯度下降法对参数进行优化，对于所有训练标签中取值

为-100 的位置，不计算损失函数并且在对参数进行梯度下降优化的过程中不参与梯度的计算，只对训练标签取值不为-100 的位置计算损失函数，通过这一方式在约 200MB 的外部金融新闻文本上进行进一步的领域自适应预训练，使得模型能够将原本拟合的适应多个主题的语言分布调整到金融文本的语言分布上来，从而能够在后续处理金融新闻舆情抽取与识别任务中获得更好的表现。

第三节　对比与分析

一、验证流程

本节将使用上文在模型介绍中提到的各类机器学习模型和深度学习模型在人工标注的一万三千条新闻舆情识别文本上进行验证和测试，具体流程如下：

首先将一万三千条有情绪标注新闻文本平均划分为十折，取其中一折为测试测试集，两折为验证集，剩余部分为训练集；

然后对于机器学习模型和非 BERT 的深度学习模型，分别使用不同的处理流程：

对于机器学习模型，首先使用 jieba 分词将语句划分为词，然后通过统计词频和计算文本特征向量 TF-IDF 的方式得到句子或者本条新闻的句向量，然后将句向量作为输入特征，送入朴素贝叶斯分类器、逻辑回归分类器和支持向量机分类器进行文本情绪识别，计算其输出作为情绪分数。

对于非 BERT 的深度学习模型，在使用 jieba 分词将语句划分为词之后，引入外部训练好的词向量作为模型 embedding 层的参数初值，然后在训练过程中对这部分参数进行同步的更新，使用模型输出的新闻舆情在{消极，中性，积极}上的分布概率作为情绪识别的结果。

对于 BERT 模型，BERT 模型由于其属于预训练模型，必须通过预先给定的词典进行分词和映射，并在句子前后分别加入[CLS]和[SEP]的特殊符号标识句子的起始位置，使用其[CLS]位置的特殊 token 输出的嵌入向量作为特征向量进行分类，并输出新闻在{消极，中性，积极}上的分布概率。

在训练过程中，使用交叉熵损失函数作为目标函数，在训练集上进行训练之后，在验证集上进行计算相关评价指标并进行验证，根据验证集上的评价指标进行参数调整，并最终选择相对较优的参数在预测集上计算其最终的情绪识别与

抽取的准确率等指标。

为了模型预测结果的稳定性，本文采用 5 折验证的方式，即在上述过程完成之后，记录评价指标，并选取未被选中的两折数据作为新的测试集，然后使用其他数据按照一折验证集七折训练集的方法进行划分，最终通过五折验证能够得到 5 组评价指标,对这些评价指标取平均最终得到最后的模型预测结果和评估。

二、模型的评价指标

本文采用准确率和 F1 分数两个评价指标作为模型评价的标准

（一）准确率

准确率：由于本文的情绪识别数据集为三分类数据集，本文采用子集准确率进行评估，即计算模型输出的标签恰好和测试集中的标签相等的个数占整个测试集样本数的比例：

$$\text{Accuracy} = \frac{N_{pred}}{N_{total}} \tag{3.7}$$

其中Accuracy为准确率取值，N_{pred}表示模型输出的标签恰好和测试集中的标签相等的个数，N_{total}表示测试集的样本数。

准确率是分类任务最常用的指标，但是这一指标在样本存在不均衡的情况下不能合理的反应模型的预测能力，特别是在模型表现较差的情况下，仅仅关注准确率则可能会出现模型将所有标签划分为一类或者两类的情况，而在本文涉及的多分类任务中，标签存在一定的不均衡情况，而不同模型由于其使用算法不同也可能存在性能上的较大差距，所以本文还采用了 F1 作为评价指标的补充。

（二）F1 score

F1 分数是统计学中用于衡量二分类或多分类中模型精确度的指标，由于其同时兼顾了分类模型的查准率和召回率，往往被用于评价样本不平衡情况下的模型表现，而在样本完全平衡的情况下，该指标退化成准确率指标。

F1 分数通过查准率和召回率得到，首先定义查准率和召回率的组成概念如下

TP：模型预测为正且实际标签为正的样本个数

TN：模型预测为负且实际标签为负的样本个数

FP：模型预测为正但实际标签为负的样本个数

FN：模型预测为负但实际标签为正的样本个数

查准率：查准率表示在所有模型预测为正的样本中，实际为正的样本数量所占的比例，即

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.8}$$

查准率越高则表示模型越不容易误把实际标签为正的样本分类成负，在进行分类模型衡量的过程中，也可以用于衡量模型判别的保守程度

召回率：召回率表示在所有实际标签为正的样本中，模型判断该标签为正的数量所占的比例，即

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.9}$$

召回率越高则表明模型越不容易漏判实际标签为正的样本，在分类模型衡量的过程中，这一指标往往用于衡量模型的严格程度。

F1 分数则通过计算查准率和召回率的加权调和平均数得到，即

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.10}$$

由于本文采用的情绪识别数据集中，标签数目为 3，在计算 F1 分数的时候采用 Macro-F1 进行计算，即将三个类别分别设置为正，并将其余两个类别设置为负，分三次统计各个类别的 TP、FP、FN、TN，然后将各类别计算得到的 TP、FP、FN、TN 加和并计算 Macro-Precision 和 Macro-Recall，最后按照上述公式计算得到 Macro-F1 作为最终评价指标。

三、模型对比结果与对比分析

评估结果如下表所示：

表 3.3　模型评估结果

模型	Accuracy	F1
朴素贝叶斯	0.6825	0.6524
TF-IDF+逻辑回归	0.6877	0.6638
TextCNN	0.6996	0.6826
TextRCNN	0.7009	0.6964
AttentionLSTM	0.7054	0.6942
BERT	0.7523	0.7289
BERT+金融文本预训练	**0.7602**	**0.7357**

根据上表所示，可以发现基于词频进行特征提取和利用的机器学习方法在处理情绪抽取与识别任务上的表现要弱于深度学习模型，这一结果在一定程度上符合自然语言处理的内部逻辑，即基于词频的词袋模型使用词或者字之间相互独立的假设，在进行分类的时候只关注词的统计特征而忽略了词法、句法和语法相关的内容，导致这类模型无法更加深刻的理解语义，而仅仅是停留在语言的表层，这些缺陷致使机器学习模型在进行情绪识别与分类的过程中忽略了过多的信息从而导致模型表现欠佳。

在深度学习模型中，TextCNN、TextRCNN 和 AttentionLSTM 这类非预训练语言模型的表现相近，其中 TextCNN 模型在稍弱于 TextRCNN 和 AttentionLSTM 模型，其原因在于，TextRCNN 和 AttentionLSTM 两个模型都使用了循环神经网络模型做实现上下文的内容的编码和抽象，与之相比的是，TextCNN 模型则仅仅通过一维卷积神经网络的方法抽取了 n 元语法特征，从抽取上下文的角度上来看，n 元语法特征依然属于局部特征，从而导致模型理解全局语义的能力较差，且信息连贯性相对割裂，最终导致 TextCNN 相对弱于 TextRCNN 和 AttenLSTM 模型。而在所有深度学习模型中，基于非预训练方法的深度学习情感抽取和识别模型在两个评价指标的表现上都弱于基于预训练的 BERT 模型，而使用外部金融文本数据进行进一步预训练的 BERT 模型在相比起没有经过外部金融文本做进一步预训练的模型又有接近 1%的提升，这种情况也是符合深度学习的规律的，即预训练的过程相当是给模型增加先验知识的过程，相比起随机初始化参数的 TextCNN、TextRCNN 和 AttentionLSTM，BERT 模型由于在大量外部文本上进行了预训练，其参数中在一定程度上包含了先验的词法、语法等相关的知识，从而能够面对在情绪抽取与识别的分类任务时有较好的先验，并得到在当前的观测空间下相对

准确性更高并且泛化能力更好的模型。

根据上表结果，本文选择在情绪识别数据集上表现最好的模型，即使用金融文本在原有的 BERT 模型上进行进一步预训练的 BERT 模型作为后续金融新闻舆情指标构建的预测模型。

第四节　金融新闻舆情指标的构建

本文基于前文所验证的使用金融文本进行进一步预训练的 BERT 模型（后文简称 FinBERT），在新闻舆情抽取和识别数据集上进行训练之后，在 2017.1.1 到 2020.6.30 号的所有涉及个股新闻上进行预测，对于每一条金融新闻，将其输入到 FinBERT 模型中并预测其情绪在积极、中性、消极三类指标上的得分，并计算单条新闻i的情绪指标如下：

$$Sent_i = pos_i - neg_i \tag{3.11}$$

即对于每一条新闻计算其积极得分超越消极得分的值，这个指标的绝对值表示了情感的极性，而该指标本身也可以被理解为新闻相对情绪程度$Sent_i$，然后对于一个横截面t上的个股，平均当日涉及到该股票的所有新闻相对情绪程度$Sent_i$，即

$$StockSentiment_{i,t} = \frac{1}{K}\sum_{k=1}^{K} Sent_{i,t,k} \tag{3.12}$$

通过对该情绪程度在时间序列上进行以十四个自然日为周期的平均，并乘以新闻与个股相关性指标$\text{Relavance}_{i,t}$，可以计算得到最终在每个横截面t上每只股票i的情绪取值

$$X_{i,t} = \text{Relavance}_{i,t} \times \left(\frac{1}{14}\sum_{i=0}^{13} StockSentiment_{i,t}\right) \tag{3.13}$$

在本指标的计算过程中，对于那些没有被报道过的股票，为了合理性考虑，将使用 0 进行空值填充，这一填充方式也符合金融新闻舆情指标的定义，即没有被报道和中性报道在情绪上应当使没有特别差异的。在指标计算结束之后，将所有十四天内没有新闻报道的个股的指标取值设置为空值。

在计算得到金融新闻舆情指标的原始值之后，本文将对金融新闻舆情指标进行去极值和标准化来计算在每个横截面上各个股票的金融新闻舆情指标取值。

本文采用中位数去极值法处理原始金融新闻舆情指标中的极端值，具体方法为：

$$\widetilde{X_{i,t}}\begin{cases}X_{mdian,t}+n*D_{MAD}, if\ X_{i,t}>X_{mdian,t}+n*D_{MAD}\\X_{mdian,t}-n*D_{MAD}, if\ X_{i,t}<X_{mdian,t}-n*D_{MAD}\\X_{i,t}\end{cases} \tag{3.14}$$

其中$\widetilde{X_{i,t}}$为去除极值之后的金融新闻舆情指标，$X_{mdian,t}$为当前横截面上原始金融新闻舆情指标的中位数取值，n是人工指定的参数，用于调整极值范围，D_{MAD}是使用当前截面上所有样本在金融新闻舆情指标上的取值减去其中位数取值的绝对值的中位数，即序列$|X_{i,t}-X_{mdian,t}|$序列的中位数，通过这一方法能够减少异常值的存在，从而减少异常值给后续研究带来的影响。

然后对金融新闻舆情指标采用 z-score 方法标准化，即对每一期金融新闻舆情指标减去该期均值之后除以该期标准差得到标准化后的金融新闻舆情指标：

$$X_{zscore}=\frac{\tilde{X}_{i,t}-mean_i(\tilde{X}_{i,t})}{std_i(\tilde{X}_{i,t})} \tag{3.15}$$

除此之外，为了剔除行业和市值对金融新闻舆情指标的影响，本文对去极值后得到的金融新闻舆情指标进行市值行业中性化，即使用申万一级行业哑变量和对数市值变量对因子做回归并取残差$\varepsilon_{i,t}$作为最终的因子取值。

$$X_{zscore,i,t}=\beta_M\ln(Mkt_{i,t})+\sum_j^n\beta_j Industry_i+\varepsilon_{i,t} \tag{3.16}$$

第四章　金融新闻舆情对股票收益率影响的实证分析

第一节　数据来源与研究设定

一、数据来源

本文使用的所有与个股相关的新闻数据来源于新浪财经、网易财经、搜狐网等等一系列互联网新闻平台中财经板块，由第四届西部量化大赛主办方收集并整理，本文只使用主办方给出的新闻文本和与新闻文本相关的个股集合与相关性数据进行研究，数据起止日期为 2017 年 1 月 1 日到 2020 年 6 月 30 日，数据为日频数据，由于新闻文本数据的起止日期限制，本文也采用 2017 年 1 月 1 日到 2020 年 6 月 30 日的行情数据进行处理，行情数据来自 tushare。

二、因子分析框架的研究设定

为了在理论和实践上研究金融新闻舆情指标与股票市场收益率之间的关系，采用 Qian 在 2004 年提出的单因子分析框架，使用基于投资组合排序法的研究和检验方式如下：

本文将利用金融新闻舆情指标作为股票排序的变量，具体流程如下：

首先确定股票池为当期所有金融新闻舆情指标有取值的股票，然后将股票池中全部股票在截面时刻 t 上按照排序变量，即金融新闻舆情指标，进行从小到大进行排序。

按照排名高低将全部股票分为 5 组，分别模拟做多五组股票作为五个资产组合，对每组内部股票采用等权重的方式进行加权，从而能够构建五组按照金融新闻舆情指标进行排序的资产组合。

由于个股在金融新闻舆情指标变量上的取值会随着事件和不同新闻报道的出现而发生一定变化，因此需要定期对上述组合进行更新，本文采用 5 日作为更新频率，更新后在此计算 5 组收益率，如此可以得到金融新闻舆情指标的收益率时间序列。

为了检验检验使用金融新闻舆情指标作为排序变量得到的 5 个组合之间的

收益率是否具有单调性，在这里本文使用分组收益率和排序变量分组的秩相关系数作为评价指标，即计算排序变量分组的高低排位为$X_g \in \{1,2,3,4,5\}$，各组的收益率高低排位为$X_r \in \{1,2,3,4,5\}$，Spearman 秩相关系数计算如下：

$$\rho_s = \frac{cov(X_r, X_g)}{\sigma_{X_r}\sigma_{X_g}} \tag{4.4}$$

当收益率随变量分组单调递减时，两者的秩相关系数值为-1，当收益率随变量分组单调递增时，秩相关系数的取值为 1。

为了能够更加充分地说明基于金融新闻舆情指标的策略有效性，本文还将使用金融新闻舆情指标构建多头选股策略，每次调仓选择舆情指标前五分之一的股票买入并持有，并利用中证 500 指数进行对冲，通过计算该对冲组合的累计净值、年化收益率、夏普比率、最大回撤等指标衡量策略有效性。

为了研究金融新闻舆情指标对股票市场收益率的预测能力，本文使用信息系数（IC，Information Coefficient）作为评价指标之一，通过计算 t 时刻的金融新闻舆情指标$X_{i,t}$和 t+1 时刻的股票收益率$R_{i,t+1}$在截面上的斯皮尔曼秩相关系数来计算和评价各期金融新闻舆情指标的表现，计算方法为：

$$\text{RankIC} = \text{spearmanr}(X_{i,t}, R_{i,t+1}) \tag{4.5}$$

信息系数衡量了预测金融新闻舆情指标所含的对未来收益率预测提供信息的信息含量，在实践中一般 IC 的绝对值高于 2%即为具有较好预测能力的指标，为了研究金融新闻舆情指标的预测能力稳定性、预测能力强弱和显著性水平，本文还将计算信息比率 IR、IC 序列绝对值大于 2%的比例和 IC 均值的 t 值来对上述评价指标进行衡量。

除了信息系数之外，个股金融新闻舆情指标与个股下一期收益率在横截面上进行普通 OLS 回归分析的 R 平方也是能够用于评价指标对个股下期收益率预测能力的重要指标

$$R_{i,t+1} = \alpha_{i,t} + \beta_t X_{i,t} + \varepsilon_{i,t} \tag{4.6}$$

第二节　金融新闻舆情指标对择股能力实证结果

一、分组回测实证分析

使用千分之三交易成本、剔除涨跌停板的分组回测结果如下图与下表所示

表 4.1　分组回测绩效评价

调仓频率	指标名称	Group1	Group2	Group3	Group4	Group5
1D	年化收益率	-38.53%	-26.45%	-25.50%	-17.38%	-4.3%
	夏普比率	-1.975	-1.221	-1.183	-0.740	-0.101
	最大回撤	82.86%	68.19%	66.35%	55.72%	43.90%
5D	年化收益率	-13.44%	-7.19%	-6.44%	-0.51%	15.94%
	夏普比率	-0.501	-0.295	-0.271	-0.021	0.741
	最大回撤	57.07%	48.32%	46.26%	39.52%	30.83%

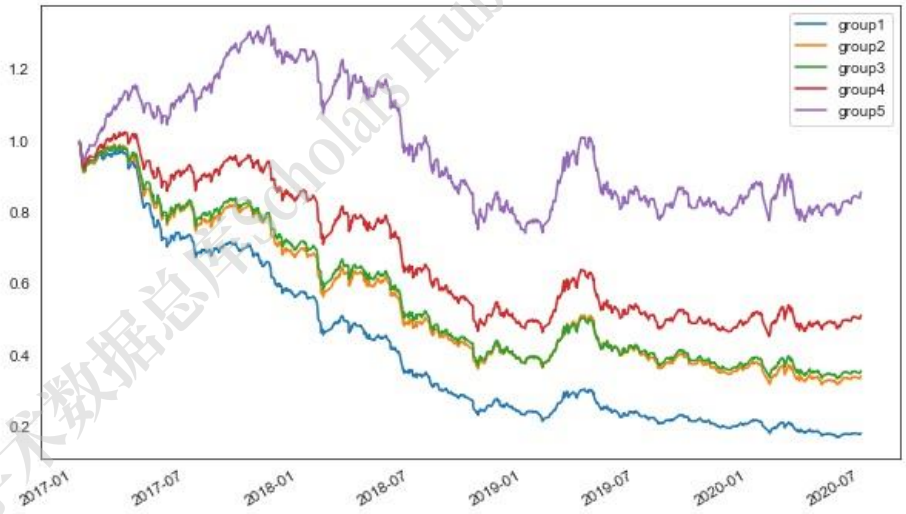

图 4.1　逐日调仓分组收益净值曲线

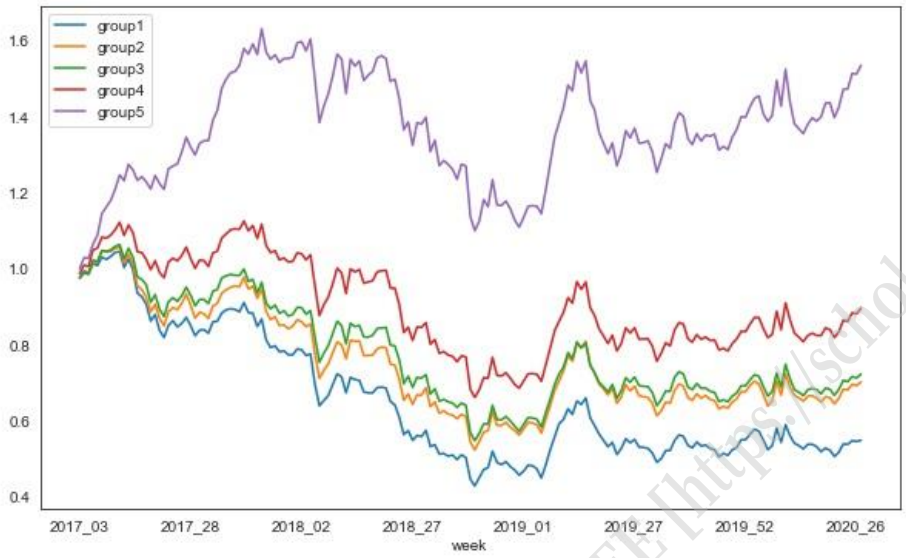

图 4.2　五日调仓分组收益净值曲线

我们将模型分为五组进行回测，分别按照每日调仓和每五日调仓的频率调整仓位,根据回测结果可以发现，根据金融新闻舆情指标划分的五组回测曲线区分度相对较高，特别是第一组和第五组之间具有显著的差异，每日调仓第五组的净值在 2020 年 6 月 30 日时为 0.8572，与之对应的第一组的净值在相同时刻为 0.1821，即如果每次交易都选择金融新闻舆情指标在前 20%的股票，三年半以来的年化收益率为-4.3%，而如果投资者每次都选择金融新闻舆情指标在后 20%的股票，三年半以来的年化收益率为-38.53%，这种情况的发生是由于回测中采用逐日调仓的方法带来了较大的换手率，收益中的较大部分被交易成本抵消；每五日调仓第五组的净值在 2020 年 6 月 30 日时为 1.6783，与之对应的第一组的净值在相同时刻为 0.6033，即如果每次交易都选择金融新闻舆情指标在前 20%的股票，三年半以来的年化收益率为 15.94%，而如果投资者每次都选择金融新闻舆情指标在后 20%的股票，三年半以来的年化收益率为-13.44%，可以发现，金融新闻舆情指标取值最高新闻报道相对更加积极的股票在之后确实会有相对较高的收益，而被消极新闻报道的股票在股票收益率的表现上则存在较大下跌的可能。除此之外还值得关注的一点是，除了取值在最大最小的两个组之外，第二三四组的表现区别相对较小，特别是第二三组之间的差异很小，这在一定程度上说明了两点：第一，由于金融新闻舆情指标采用过去十四天的指数加权平均进行

构造，即距离相对当前时间较远的新闻对未来一段时间的股票收益率的预测能力会有所下降，这在一定程度上表明了涉及个股新闻的时效性，作为投资者在利用和研究金融新闻舆情指标的时候应当充分考虑时效性带来的影响；第二，具有显著情绪倾向的新闻能够对股票市场收益率具有相对更强的预测能力，即大量持续新闻正向情绪或者负面情绪会通过新闻媒体传播给其新闻报道的接收方，接收方会因为积极或消极内容和新闻舆情信息的大量涌入而对其自身的投资行为产生一定的调整，并最终反映在股价上。根据上表可以发现，每五日调仓和每日调仓在分组年化收益率，夏普比率和最大回撤三个指标上都和每日调仓具有类似的组内排序和趋势，但每五日调仓相对于每日调仓的年化收益率更高，夏普比率也相对更高，其原因在于调仓频率的降低带来了换手率的降低，从而降低了交易成本，而交易成本的降低最终带来了较高的收益率；除此之外，通过上述对比也可以发现金融新闻舆情指标在区分下一期个股收益率的能力上较强，而倾向性更强的情绪在区分能力上尤为突出。

除了分组回测的收益率，本文计算其五组之间的收益率和组别排序的斯皮尔曼秩相关系数为 1，可以说明金融新闻舆情指标的大小和其下一期股票市场收益率具有较好的相关性。

二、策略与基准对比分析

除了分组回测，由于我国股票市场对做空具有一定的限制，本文通过对比基于金融新闻舆情指标的纯多头策略并与中证 500 指数进行对比，并根据上文结果选择五日作为调仓周期，交易成本为千分之三并过滤涨跌停板，其收益率曲线和相关指标计算如下图所示：

表 4.2　组合绩效对比

	年化收益率	最大回撤	夏普比率	累计净值
多头组合	15.94%	30.83%	0.741	1.6783
中证 500	-1.47%	38.47%	-0.0691	0.9495

图 4.3　组合净值曲线对比

上图中 long 曲线代表通过对新闻舆情进行准确识别并买入持有每期金融新闻舆情指标前 20%的股票所构建的投资组合，而 benchmark 曲线则代表买入并持有中证 500 指数的组合收益，可以发现持有多头组合的年化收益率为 15.94%，高于持有以中证 500 指数为代表的市场组合，通过对比夏普指数可以发现，多头组合的夏普比率约为 0.741，也高于持有中证 500 的夏普比率-0.0691，通过在年化收益率、最大回撤、夏普比率等评价上的对比可以发现，使用金融新闻舆情指标构建的投资组合在各个评价指标上都优于持有中证 500 指数。通过对比金融新闻舆情指标投资组合和市场组合，可以发现金融新闻舆情指标在预测能力上的有效性。

为了研究舆情指标组合能否获取超越市场收益率的超额收益，使用中证 500 指数对冲金融新闻舆情指标组合，计算相关指标并画图如下：

表 4.3　对冲组合绩效

	年化收益率	最大回撤	夏普比率	累计净值
对冲组合	17.19%	2.843%	2.776	1.7425

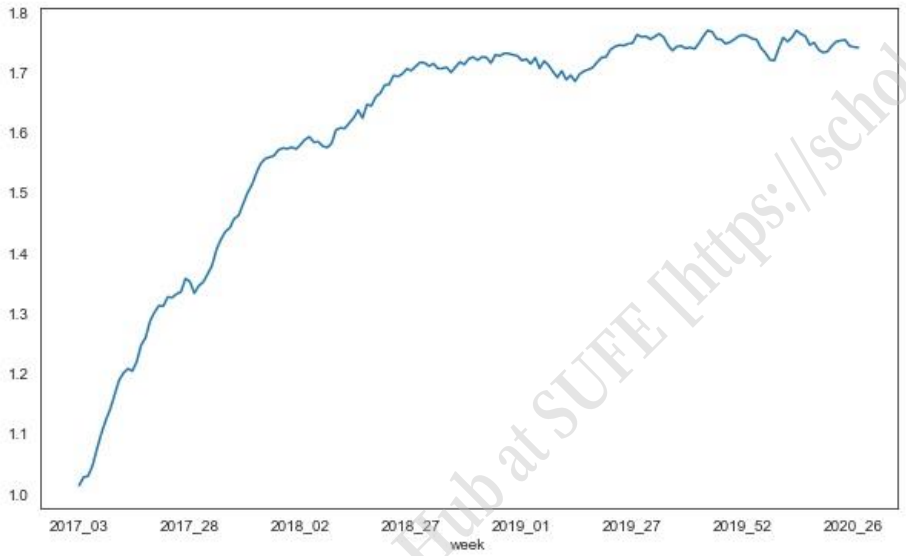

图 4.4　对冲组合净值曲线

使用中证 500 指数对金融新闻舆情指标组合后，对冲组合在最大回撤为 2.843%的情况下实现了年化 17.19%的收益率水平，并在 2020 年上半年达到的累计净值 1.7425，通过观察收益率曲线可以发现，对冲组合在 2017 年初到 2018 年上半年期间具有良好的收益率表现，而在 2018 年下半年开始，收益水平逐渐趋于平缓，但波动相对较小，这也在一定程度上说明该指标在 2017 年到 2018 年上半年的预测能力良好，而随着信息利用效率的提高和新闻等另类数据逐渐被引入投资领域，指标的收益率预测能力在 2019 年后有所下降。

三、指标预测能力实证分析

在 2017 年到 2020 年上半年计算信息系数并得到如下结果：

表 4.4　金融新闻舆情指标 RankIC 统计指标

Year	Count	Mean	Std	Min	25%	50%	75%	Max
2017	239.0	0.0277	0.0561	-0.129	-0.011	0.023	0.062	0.177
2018	243.0	0.0179	0.0688	-0.218	-0.023	0.016	0.060	0.193
2019	244.0	0.0145	0.0663	-0.215	-0.022	0.018	0.053	0.215
2020	118.0	0.0195	0.0673	-0.145	-0.022	0.018	0.059	0.308
total	844	0.01526	0.0644	-0.203	-0.026	0.013	0.055	0.286

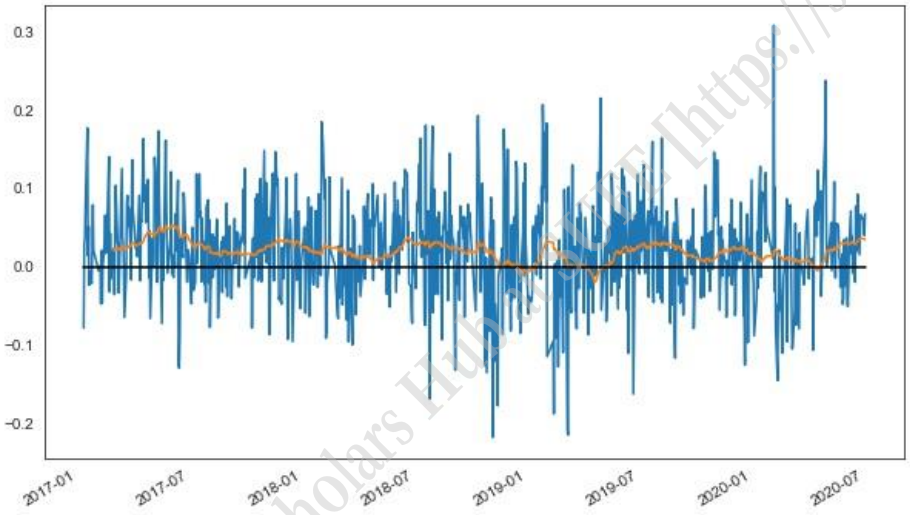

图 4.5　金融新闻舆情指标 RankIC 日度序列

为了研究金融新闻舆情指标在 2017-2020 年上半年每个时期的表现，我们分别计算金融新闻舆情指标在 2017、2018、2019 和 2020 年以及整个回测期间的描述性统计指标，并绘制金融新闻舆情指标于横截面股票收益相关系数图如上，其中在 2017 年的 239 个交易日中，金融新闻舆情指标与股票市场横截面收益率之间相关性为正的有 161 个，相关性为负的有 78 个；2018 年的 243 个交易日中，金融新闻舆情指标与股票市场横截面收益率之间相关性为正的有 149 个，相关性为负的有 94 个；2019 年的 244 个交易日中，金融新闻舆情指标与股票市场横截面收益率之间相关性为正的有 152 个，相关性为负的有 92 个；2020 年上半年的 118 个交易日中，金融新闻舆情指标与股票市场横截面收益率之间相关性为正的有 72 个，相关性为负的有 46 个。

根据上表可以发现，金融新闻舆情因子在 2017 年和 2020 年上半年这两个时间段具有较好的表现，其秩相关系数的均值分别可以达到 2.7%和 1.95%，而其在 2018 年和 2019 年的表现相对相对弱于其在 2017 和 2020 年上半年的表现，但仍然高于 1%，值得关注的是，无论是在哪一年，其 RankIC 的百分之七十五分位数的取值都相对较高，而在某些特定的时间点，金融新闻舆情指标与股票市场的横截面收益的相关性会高于 20%，这在一定程度上可以说明新闻舆情，特别是具有非常显著倾向性的新闻情绪会在某些时刻对股票市场产生十分显著的影响。

为了研究金融新闻舆情指标预测能力的稳定性，预测能力的强弱和 IC 是否显著等特性，本文计算了信息比率、IC 绝对值大于 2%的比例和 IC 的 t 值等指标，通过下表可以发现，金融新闻舆情指标总体上具有较好的预测能力稳定性，信息比率 IR 达到了 0.3092，说明运用金融新闻舆情指标与下一期的股票市场横截面收益率的相关性相对比较稳定；从 2017-2020 年上半年分别来看，2017 年 IC 序列绝对值大于 2%的比例为 70.29%，2018 年 IC 序列绝对值大于 2%的比例为 76.54%，2019 年 IC 序列绝对值大于 2%的比例为 76.23%，2020 年 IC 序列绝对值大于 2%的比例为 74.58%，值得关注的一点是，在 2018 和 2019 年两年间，金融新闻舆情指标的 IC 均值相对较低但其 IC 序列绝对值大于 2%的比例则相对更高，这在一定程度上说明了金融新闻舆情指标的局限性，即在市场表现相对较差的情况下，金融新闻舆情指标对股价变动方向的影响不是完全线性的，但能够在一定程度上解释其波动。

使用金融新闻舆情指标与下期个股收益率横截面回归并计算 R 平方结果如下：

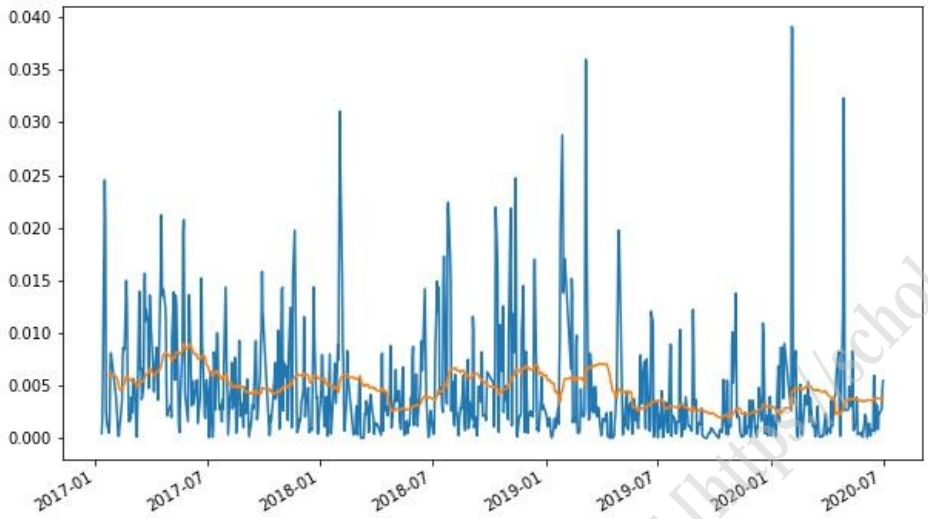

图 4.6　金融新闻舆情指标 R 平方日度序列

通过观察 R 平方日度序列的结果可以发现：金融新闻舆情指标的预测能力整体较强，在 2017 与 2018 年，R 平方序列均值能够达到 0.0054 与 0.0046，说明在 2017 到 2018 年间，金融新闻舆情指标对于下一期收益率的预测能力相对较好，而在 2019 年间 R 平方取值相对较低，但在 2020 年上半年该指标的预测能力则有所上升，总体而言，2017 年到 2020 年上半年，R 平方序列的均值为 0.0042，表明金融新闻舆情指标对个股收益率具有一定预测能力，这一结论与基于 RankIC 的评价结果相符。

第五章　结论与展望

第一节　研究结论

根据《中国互联网发展报告 2020》，截止至 2019 年底，我国互联网用户的规模已经达到 13.19 亿，网络新闻用户规模达到 7.31 亿，互联网新闻正逐渐成为用户获取外界信息的重要途径，互联网金融新闻媒体通过对我国宏观经济形势、产业政策和市场与企业新闻内容的报道，成为使得投资者可以实时、快捷获取市场信息的重要途经和媒介，而这类新闻可能会在一定程度上影响投资者的行为操作和企业管理者的决策，从而对股票市场收益率产生影响，因此，通过先进的手段对金融新闻进行文本挖掘和提取出新闻中包含的情绪具有现实和理论的双重意义。

为了解决该领域标注样本较少的问题，本文通过人工标注和外界样本混合的方式获取到一万三千条金融新闻，并将新闻舆情分为消极、中性和积极三类，由此构建了新闻舆情识别数据集，之后在新闻舆情识别数据集上研究了传统机器学习和基于神经网络模型的深度学习方法在识别情绪准确性上的表现，研究表明使用基于 Bag of words 的假设，利用词频方法对文本提取信息并使用传统机器学习模型进行情绪识别的方法在 F1 和准确率两个评价指标上都显著低于基于神经网络的深度学习模型，并且基于预训练方法的 BERT 模型在深度学习模型中表现远超其他模型，具有较强的情绪识别能力。

为了进一步提升 BERT 模型的在情绪抽取与识别上的表现，本文采用外部开源的金融文本对 BERT 模型使用 MLM 方法进行了进一步的预训练，从而在一定程度上将原本开源的 BERT 模型在多个主题文本上学到的语言分布逐渐调整到偏向学习金融文本的语言分布上来，并将调整后的 BERT 模型在新闻舆情识别数据集上进行训练，并与未经过这一调整的 BERT 模型表现进行对比发现，经过调整之后的模型在情绪抽取与识别上有更好的表现。

为了验证金融新闻舆情指标在选股能力上的表现，本文分别采用了投资组合排序法和信息系数的方式来衡量涉及个股的新闻舆情对于其下一期收益率的预测能力，在投资组合排序法中，本文在每一期按照个股金融新闻舆情指标的取值大小分为五组，在每一组内等权重做多组内的股票并记录当期收益率并在下

一期按照新的金融新闻舆情指标换仓，除此之外本文也构造了多空组合来验证在资金中性的情况下资产组合的收益率与风险情况；在基于信息系数的研究中，本文计算了当期个股的金融新闻舆情指标与下期收益率的斯皮尔曼秩相关系数，并计算基于该系数的一系列指标来研究金融新闻舆情指标的预测能力。分组回测的研究结果表明金融新闻舆情指标在区分个股下期收益率上具有较好的表现，特别是极端正面或者负面的情绪能够较好的预测个股的收益率水平；通过对每年的信息系数的计算发现，金融新闻舆情指标在整体上与下期收益率的相关系数相对较高，具体到每年而言，2017 和 2020 年的金融新闻舆情指标相对 2018 和 2019 年的影响更大，而通过信息比率和 IC 大于 2%的比率计算可以发现金融新闻舆情指标的稳定性和预测能力都较强。

对于投资者而言，互联网财经媒体的新闻报道实时、有效，包含信息内容十分广泛，往往其内容涵盖了对于市场现象、公司实时情况的陈述和评价，除此之外还包含了关于上市公司重大新闻和社会上突发事件等，这些信息能够为投资者提供投资决策的参考依据，投资者密切关注新闻，掌握市场行情和上市公司信息是十分重要的，但也应充分识别新闻中包含的情绪因素，根据自身的情况做出合理准确的理性判断。

对于政府而言，由于互联网新闻所面向对象群体十分庞大，其传递出的信息对于市场会产生较大影响，为了更好的服务于投资者，避免市场受到剧烈情绪的影响，维护金融市场的稳定，政府应当建立健全相关法律法规，加大网络新闻监督力度，还应该创新金融市场舆论监管方式，采用先进的文本挖掘手段和技术更加精准的对市场情绪和新闻舆情进行识别。

第二节　不足与展望

本文也存在一定的不足之处，首先，由于实验环境的限制，本文采用的 BERT 模型是中文开源 BERT 模型中参数量最少的一个，即 HuggingFace 开源的 BERT-Base-Chinese 模型，并在此基础上实现了上述情绪识别的实验，在这一模型之上还有隐藏层数目更多、参数量更大的 BERT-Large 等模型，研究表明，这类参数量更大的模型在大量数据集上的表现都显著优于本文所使用的小参数量 BERT 模型，在实际投资和运用的过程中，通过更新设备和环境，可以得到更为准确的情绪识别与抽取能力；其次本文仅仅基于 2017 年到 2020 年上半年的股票市场

历史数据进行的回测，市场未来可能会出现较大的变化，基于量化模型的方法往往无法完全衡量市场风险，并且由于数据可得性限制，本文并未研究 2020 年下半年至今金融新闻舆情指标的表现；除此之外，本文所采用的数据是相对低频的日频数据，而没有详细研究新闻对于分钟级别行情的影响能力。最后，基于金融新闻舆情指标的选股方法实际上是建立在行为金融学中对于投资者非理性的前提假设下的，这一假设可能随着我国市场的不断发展完善、投资者教育水平、信息获取与处理能力的不断提升以及机构投资者占比的提升而被逐渐削弱，从而导致金融新闻舆情指标在未来的效果会被削弱。

对于上文提到的第一个不足之处，可以通过利用更大的模型和更多的标注数据来训练在新闻舆情抽取和识别任务上表现更佳的模型，从而尽可能减少由于模型情绪抽取错误带来的金融新闻舆情指标计算偏差；除此之外，本文为了计算方便，采用了简单平均的形式计算情绪指标，在实际运用过程中可以通过对金融新闻舆情指标进行进一步的计算和处理，更好地利用和挖掘新闻这一另类数据的信息。

参考文献

[1] 黄润鹏，左文明，& 毕凌燕. (2015). 基于微博情绪信息的股票市场预测. 管理工程学报, 29(1), 47-52.

[2] 李岩，金德环. (2017) 投资者互动与股票收益——来自社交媒体的经验证据. 金融论坛, 2017, 5: 72-80.

[3] 孟雪井，杨亚飞，& 赵新泉. (2016). 财经新闻与股市投资策略研究——基于财经网站的文本挖掘. 投资研究, 8, 29-37.

[4] 孟雪井，杨亚飞，& 赵新泉. (2016). 财经新闻与股市投资策略研究——基于财经网站的文本挖掘. 投资研究, 8, 29-37.

[5] 杨洁，詹文杰，& 刘睿智. (2016). 媒体报道，机构持股与股价波动非同步性. 管理评论, 28(12), 30.

[6] 杨继东. (2007). 媒体影响了投资者行为吗?——基于文献的一个思考. 金融研究, 11, 93-102.

[7] 郑瑶, 董大勇 & 朱宏泉. (2016). 异质性情绪影响股市羊群效应吗?——来自互联网股票社区的证据. 系统工程(09), 9-14.

[8] Antweiler, W., & Frank, M. Z. (2004). Is all that talk just noise? The information content of internet stock message boards.The Journal of finance,59(3), 1259-1294.

[9] Baker, M., & Stein, J. C. (2004). Market liquidity as a sentiment indicator.Journal of Financial Markets,7(3), 271-299.

[10] Brown, G. W. . (1999). Volatility, sentiment, and noise traders.Financial Analysts Journal,55(2), 82-90.

[11] Brown, G. W. , & Cliff, M. T. . (2004). Investor sentiment and the near-term stock market.Journal of Empirical Finance,11(1), 1-27.

[12] Wurgler, J. A. , & Baker, M. P. . (2006). Investor sentiment and the cross-section of stock returns.Economic Management Journal,61(4), 1645-1680.

[13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need.arXiv preprint arXiv:1706.03762.

[14] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.arXiv preprint arXiv:1810.04805.

[15] Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., & Smith, N. A. (2020). Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. arXiv preprint arXiv:2004.10964.

[16] Huang, D., Jiang, F., Tu, J., & Zhou, G. (2015). Investor sentiment aligned: A powerful predictor of stock returns.The Review of Financial Studies,28(3), 791-837.

[17] Hillert, A., Jacobs, H., & Müller, S. (2014). Media makes momentum. The Review of Financial Studies,27(12), 3467-3501.

[18]Ibbotson, R. G., Sindelar, J. L., & Ritter, J. R. (1994). The market's problems with the pricing of initial public offerings. Journal of applied corporate finance,7(1), 66-74.

[19]Joachims, T. (1998, April). Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning (pp. 137-142).

[20]Loughran, T., & McDonald, B. (2016). Textual analysis in accounting and finance: A survey. Journal of Accounting Research, 54(4), 1187-1230.

[21]Lee, W. Y., Jiang, C. X., & Indro, D. C. (2002). Stock market volatility, excess returns, and the role of investor sentiment. Journal of banking & Finance,26(12), 2277-2299.

[22]Ljungqvist, A. ,Nanda, V. , & Singh, R. . (2006). Hot markets, investor sentiment, and ipo pricing.The Journal of Business,79(4), 1667-1702.

[23]Maron, M. E. . (1961). Automatic indexing: an experimental inquiry. Journal of the Acm,8(3), 404-417.

[24]Rognone, L., Hyde, S., & Zhang, S. S. (2020). News sentiment in the cryptocurrency market: An empirical comparison with Forex. International Review of Financial Analysis,69, 101462.

[25]Siegel, J. J. (1992). Equity risk premia, corporate profit forecasts, and investor sentiment around the stock crash of October 1987.Journal of Business, 557-570.

[26]Tetlock, P. C. (2007). Giving content to investor sentiment: The role of media in the stock market. The Journal of finance,62(3), 1139-1168.

[27]Tetlock, P. C., Saar-Tsechansky, M., & Macskassy, S. (2008). More than words: Quantifying language to measure firms' fundamentals. The Journal of Finance,63(3), 1437-1467.

[28]Wei, Y. C., Lu, Y. C., Chen, J. N., & Hsu, Y. J. (2017). Informativeness of the market news sentiment in the Taiwan stock market. The North American Journal of Economics and Finance,39, 158-181.

[29]Zhang, X., Fuehres, H., & Gloor, P. A. (2011). Predicting stock market indicators through twitter “I hope it is not as bad as I fear”. Procedia-Social and Behavioral Sciences,26, 55-62.

附录

用于预训练 BERT 和训练 BERT 的代码如下

pretrain_utils.py

```
import random
import math

import torch
import pandas as pd
import numpy as np
from transformers import BertModel, BertTokenizer
from torch.utils.data import Dataset
from torch.optim.lr_scheduler import LambdaLR

class PretrainPreprocessor(object):
    """
    process input sentence(tokenization, MLM task) in for PretrainDataset
    """
    def __init__(self, bert_dir, max_len=100):
        self.bert_dir = bert_dir
        self.max_len = max_len

        self.tokenizer = BertTokenizer.from_pretrained(bert_dir)


    def tokenize(self, sentence):
        tokenized_output = self.tokenizer.encode_plus(sentence,
pad_to_max_length=True, max_length=self.max_len)
        return tokenized_output
```

```
        def mlm(self, tokenized_output):
            input_ids = tokenized_output["input_ids"]
            attention_mask = tokenized_output["attention_mask"]
            output_ids = list()
            mlm_labels = list()
            for i, token in enumerate(input_ids):
                should_mask = random.random() > 0.7
                mask_method_rand = random.random()
                use_mask_token = mask_method_rand < 0.7
                use_replace_token = mask_method_rand > 0.7 and
mask_method_rand < 0.85
                use_ori_token = mask_method_rand > 0.85
                if attention_mask[i] == 1:
                    if should_mask:
                        mlm_labels.append(token)
                        mask_token_id = self.tokenizer.mask_token_id
                        if use_mask_token:
                            output_ids.append(mask_token_id)
                        elif use_replace_token:
                            output_ids.append(

random.sample(list(self.tokenizer.vocab.values())[200:], k=1)[0]
                            )
                        else:
                            output_ids.append(token)
                    else:
                        mlm_labels.append(-100)
                        output_ids.append(token)
                else:
                    output_ids.append(self.tokenizer.mask_token_id)
                    mlm_labels.append(-100)
```

```
            return output_ids, mlm_labels


    class PretrainDataset(Dataset):
        """
        Dataset for pretrain BERT
        """
        def __init__(self, bert_dir, pretrain_dataset_path, max_len=100):
            super(PretrainDataset, self).__init__()
            self.bert_dir = bert_dir
            self.pretrain_dataset_path = pretrain_dataset_path
            self.max_len = max_len

            self.preprocessor = PretrainPreprocessor(bert_dir, max_len=max_len)
            self.pretrain_dataset = pd.read_csv(pretrain_dataset_path)


        def __getitem__(self, item):
            line = self.pretrain_dataset_path.iloc[item]
            sentence = line["title"]
            tk_output = self.preprocessor.tokenize(sentence)
            output_ids, mlm_labels = self.preprocessor.mlm(tk_output)
            return output_ids, tk_output["token_type_ids"],
tk_output["attention_mask"], mlm_labels


    class ConstantLRWithWarmup(LambdaLR):
        """
        Increase lr to optimizer.lr linearly with ratio cur_step/num_warmup_steps
        If cur_step >= num_warmup_steps, the ratio will be 1.0
```

```
        Example:
            >>>  scheduler  =  ConstantLRWithWarmup(optimizer,
num_warmup_steps, last_epoch)
            ...
            >>> loss.backward()
            >>> optimizer.step()
            >>> scheduler.step()

        """
        def __init__(self, optimizer, num_warmup_steps, last_epoch=-1):
            def lr_lambda(cur_step):
                if cur_step < num_warmup_steps:
                    return float(cur_step) / float(max(num_warmup_steps, 1.0))
                return 1.0
            super(ConstantLRWithWarmup,  self).__init__(optimizer,  lr_lambda,
last_epoch)
```

pretrain_bert.py

```
import torch
from torch import nn
import numpy as np
from transformers import BertForMaskedLM
from pretrain_utils import PretrainDataset, ConstantLRWithWarmup
from collections import deque

torch.cuda.empty_cache()
BATCH_SIZE = 32
LR = 2e-5
EPOCH = 5
MAXLEN = 140
DISPLAY = 35
```

```
BERT_PATH = 'YOUR_BERT_PATH'
FILE_PATH = 'YOUR_FILE_PATH'
device = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')

dataset = PretrainDataset(
    file_path=FILE_PATH,
    bert_path=BERT_PATH,
    maxlen=MAXLEN
)
dataloader = torch.utils.data.DataLoader(dataset, batch_size=BATCH_SIZE)
model = BertForMaskedLM.from_pretrained(BERT_PATH).to(device)
optimizer = torch.optim.AdamW(model.parameters(), lr=LR)
scheduler = ConstantLRWithWarmup(optimizer, num_warmup_steps=10000)

optimizer.zero_grad()
optimizer.step()

for e in range(EPOCH):
    losses = deque([], maxlen=DISPLAY)
    for i, (input_ids, token_type_ids, attention_mask, mlm_labels) in enumerate(dataloader):
        input_ids, token_type_ids, attention_mask, mlm_labels = list(
            map(lambda x:x.to(device), [input_ids, token_type_ids, attention_mask, mlm_labels]))
        output = model(input_ids, attention_mask, token_type_ids, masked_lm_labels=mlm_labels)
        loss = output[0]
        losses.append(loss.item())

        optimizer.zero_grad()
        loss.backward()
```

```
            optimizer.step()
            scheduler.step()

            if i != 0 and i % DISPLAY == 0:
                print('step: {}\tcurrent loss: {}'.format(i,np.mean(losses)))

torch.save(model.bert, './embeddings/after_pretrain_{}.pt'.format(e))
```

train_utils.py

```
import torch
import pandas as pd
import numpy as np
from torch.utils.data import Dataset
from transformers import BertTokenizer
from torch.optim.lr_scheduler import LambdaLR

def to_device(inputs, device='cuda:0'):
    if isinstance(inputs, (list, tuple)):
        outputs = []
        for i, item in enumerate(inputs):
            outputs += to_device(item, device)
    elif isinstance(inputs, (torch.Tensor, torch.nn.Module)):
        return inputs.to(device)
    elif isinstance(inputs, dict):
        outputs = {}
        for key in inputs.keys():
            outputs[key] = to_device(inputs[key], device)
    else:
        outputs = inputs
    return outputs
```

```
class ClassificationDataset(torch.utils.data.Dataset):
    """Some Information about MyDataset"""
    def __init__(self, data, bert_dir="../bert_weight/raw", max_len=50):
        super(ClassificationDataset, self).__init__()
        self.max_len = max_len
        self.data = data
        self.tokenizer = BertTokenizer.from_pretrained(bert_dir)

    def __getitem__(self, index):
        line = self.data.iloc[index]
        sentence, label = line.title, line.label
        tk_output = self.tokenizer.encode_plus(sentence,
max_length=self.max_len, pad_to_max_length=True)
        input_ids = torch.LongTensor(tk_output["input_ids"])
        attention_mask = torch.LongTensor(tk_output["attention_mask"])
        token_type_ids = torch.LongTensor(tk_output["token_type_ids"])
        label = torch.LongTensor([label])
        return input_ids, attention_mask, token_type_ids, label

    def __len__(self):
        return len(self.data)

class ConstantLRWithWarmup(LambdaLR):
    """
    Increase lr to optimizer.lr linearly with ratio cur_step/num_warmup_steps
    If cur_step >= num_warmup_steps, the ratio will be 1.0
    Example:
        >>> scheduler = ConstantLRWithWarmup(optimizer,
num_warmup_steps, last_epoch)
        ...
        >>> loss.backward()
```

```
            >>> scheduler.step()
            >>> optimizer.step()
        """
        def __init__(self, optimizer, num_warmup_steps, last_epoch=-1):
            def lr_lambda(cur_step):
                if cur_step < num_warmup_steps:
                    return float(cur_step) / float(max(num_warmup_steps, 1.0))
                return 1.0
            super(ConstantLRWithWarmup, self).__init__(optimizer, lr_lambda, last_epoch)
```

train_nn.py

```
from collections import deque

import torch
import pandas as pd
import numpy as np
from sklearn.model_selection import KFold
from sklearn.metrics import f1_score, accuracy_score

from train_utils import ClassificationDataset, to_device, ConstantLRWithWarmup

from transformers import BertForSequenceClassification


class config:
    num_epoch = 5
    batch_size = 32
    bert_lr = 5e-5
    lr = 5e-4
```

```
        device = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
        bert_dir = "YOUR_BERT_DIR"
        bert_path = "YOUR_RETRAINED_OUTPUT"
        data_path = "YOUR_CLASSIFICATION_DATASET_PATH"
        model_name = "bert"

        # seq_max_len = 50
        # emb_size = 200
        # hidden_size = 128
        # output_size = 3
        # kernels = [2,3,4]

    data = pd.read_csv(config.data_path)
    data = data[pd.isnull(data.title)==False]# .iloc[:1000, :]

    kf = KFold()
    for train_idx, test_idx in kf.get_n_splits(data):
        train_data, test_data = data.iloc[train_idx], data.iloc[test_idx]
        train_dataset = ClassificationDataset(train_data, bert_dir=config.bert_dir,
max_len=config.seq_max_len)
        test_dataset = ClassificationDataset(test_data, bert_dir=config.bert_dir,
max_len=config.seq_max_len)

        train_loader = torch.utils.data.DataLoader(train_dataset,
batch_size=config.batch_size)
        test_loader = torch.utils.data.DataLoader(test_dataset,
batch_size=config.batch_size)

        model = BertForSequenceClassification(config)
        if config.bert_path:
            model.bert.load_state_dict(config.bert_path)
```

```
        model = model.to(config.device)

        if config.model_name == "bert":
            optimizer = torch.optim.AdamW(model.parameters(), lr=config.bert_lr)
        else:
            optimizer = torch.optim.AdamW(model.parameters(), lr=config.lr)
        scheduler = ConstantLRWithWarmup(optimizer, 500)
        optimizer.zero_grad()
        optimizer.step()
        criterion = torch.nn.CrossEntropyLoss()

        for epoch in range(config.num_epoch):
            queue = deque(maxlen=100)
            model.train()
            for i, data in enumerate(train_loader):
                input_ids, attention_mask, token_type_ids, label = data
                input_ids, attention_mask, token_type_ids, label = input_ids.to(config.device), attention_mask.to(config.device), token_type_ids.to(config.device), label.to(config.device)
                y_pred = model(
                    input_ids=input_ids,
                    attention_mask=attention_mask,
                    token_type_ids=token_type_ids
                )
                loss = criterion(y_pred, label.squeeze())
                queue.append(loss.item())

                optimizer.zero_grad()
                loss.backward()
                scheduler.step()
```

```
                optimizer.step()

                if i != 0 and i % 50 == 0:
                    print("Current loss: {}".format(np.mean(queue)))

            model.eval()
            with torch.no_grad():
                y_preds = []
                y_trues = []
                for data in test_loader:
                    input_ids, attention_mask, token_type_ids, label = data
                    input_ids, attention_mask, token_type_ids, label = input_ids.to(config.device), attention_mask.to(config.device), token_type_ids.to(config.device), label.to(config.device)
                    y_pred = model(
                        input_ids=input_ids,
                        attention_mask=attention_mask,
                        token_type_ids=token_type_ids
                    )
                    y_preds += y_pred.argmax(dim=-1).tolist()
                    y_trues += label.squeeze().tolist()
                print("Current validation accuracy: {}\nCurrent validation f1: {}".format(
                    accuracy_score(y_pred=y_preds, y_true=y_trues),
                    f1_score(y_pred=y_preds, y_true=y_trues, average="macro"))
                )
```

致谢

行文至此，论文的主要内容便已经基本结束了，从论文的题目选择、资料收集到论文内容撰写和与格式的调整，这篇论文的完成离不开许多人的支持和帮助。

首先我需要感谢我的导师，她从我研究生入学开始就在学习和生活上给了我许多关注和支持，也是在我的导师和实验室诸位老师的带领和启发下，我才逐渐走上了金融科技中关于深度学习自然语言处理算法的学习和研究；导师在这篇论文的选题、内容撰写和格式调整上都给予了我很大的帮助，她严谨的治学态度和细心认真的教导，使我获益匪浅。在此，谨向我的导师致以我最衷心的感谢！

这篇论文的顺利完成，还离不开实验室各位老师和同学的支持和帮助，在实验室组会上，各位同学针对这篇论文的早期思路提出了许多值得借鉴和学习的地方，感谢各位同学在组会中的踊跃发言和和组会上各位老师对这篇论文的思路的指导与帮助。

我还需要感谢在院内评审过程中对这篇论文提出评审意见的三位老师们，感谢各位老师对这篇论文提出的修改意见，使这篇论文的内容更加充实、格式更加规范。

感谢在硕士研究生期间的同学们，感谢春节期间依然坚守在岗位的学校工作人员，最后，我要感谢我的父母多年对我在生活上的支持和各方面的培养。

个人简历及在学期间发表的研究成果

个人简历:

孙明宏，男，河北石家庄，1997 年 6 月 10 日出生

2012 年 9 月至 2015 年 6 月，高中就读于石家庄市第二中学

2015 年 9 月至 2019 年 6 月，本科就读于上海财经大学金融学院金融工程专业

2019 年 9 月至 2021 年 6 月，硕士就读于上海财经大学金融学院金融硕士专业