学校代码：10272
学　　号：2021211340

上海财经大学

SHANGHAI UNIVERSITY OF FINANCE AND ECONOMICS

硕士学位论文

MASTER　DISSERTATION

论文题目	网络新闻舆情对股指期货收益率的影响研究
作者姓名	徐英皓
院(系所)	金融学院
专　　业	金融科技
指导教师	谢斐
完成日期	2023 年 3 月 18 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：徐英皓　　日期：2023 年 3 月 20 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：徐英皓　　导师签名：

日期：2023 年 3 月 20 日　　日期：2023 年 3 月 20 日

摘要

网络舆情是指网络上公众对某个话题、事件、产品、品牌等的言论和情感倾向的总和。随着互联网的普及和社交媒体的兴起，网络舆情已成为影响社会舆论和决策的重要因素之一。网络舆情与金融市场密切相关：首先，投资者的决策往往受到网络舆情的影响，一些负面新闻、谣言等可能会引起投资者的恐慌性抛售，从而对股票价格产生负面影响。另外，网络舆情中的一些信息和预测可能会影响股票价格的走向，比如某些重要人物的言论、政策调整等，这些都可能成为投资者进行决策的重要依据。因此，对网络舆情的监测和分析，对于预测金融市场价格的变动趋势具有重要意义。近年来，随着社交媒体的不断发展和普及，越来越多的投资者开始通过社交媒体等渠道获取市场信息，这种方式与传统媒体相比更具有即时性和广泛性。因此，如何利用大数据分析技术和机器学习方法，挖掘网络舆情中的信息价值，预测金融市场价格的变动趋势，成为了当今金融领域的研究热点。

现有对于文本情感识别的研究首先包括基于手工提取的特征来训练分类器的传统机器学习情绪识别模型，这些特征需要经过专业的领域知识或者经验来确定，因此比较依赖人工经验和数据预处理的质量，同时不能充分利用大规模语料中的隐含信息，无法解决语言的歧义性和复杂性。其次是非预训练的深度学习情绪识别模型，通常使用的是神经网络从原始文本数据中学习特征并训练分类器，其特点是可以自动提取文本中的相关特征，避免了特征工程和人工特征提取的问题。然而，由于非预训练的深度学习模型需要较多的标注数据来进行训练，并且容易受到数据集的噪声影响。最后是基于预训练的深度学习情绪识别模型，其特点是可以利用大量的未标注数据进行预训练，从而获得更好的文本表示能力，具有更好的可迁移性和泛化性能。然而，预训练模型需要更大的计算资源和数据集，并且需要更多的领域专业知识来进行调整和优化。

本文通过东方财富股指期货股吧获取 117245 条网络舆情数据，并且手工标注 5863 条文本提供给 BERT 进行预训练任务以提高识别准确率，并且为了将非预训练深度学习情绪识别模型和基于预训练的深度学习情绪识别模型效果对比，选取 TextCNN、TextRNN、TextRCNN、TextRNN_Att、Transformer、BERT、BERT_CNN、BERT_DPCNN、BERT_RNN、BERT_RCNN 和 ENRIE 等 11 个模型进行训练测试，之后报告它们的 F1 得分和准确率，特别地对比了 BERT 与基于金融文本预训练后 BERT 的结果。在选出预测效果最优的模型后构建情绪指标，并计算其与 IF00、IC00 和 IH00 三个股指期货主力合约收益率的相关性。最后使用 LSTM 模型预测股指期货价格，同时对比了是否加入情绪指标对模型预测效果的影响。

本文得出的结论包括：（1）基于预训练的深度学习情绪识别模型的预测效果优于非预训练深度学习情绪识别模型。对于非预训练深度学习模型，适当增加神经网络的复杂程度对于预测效果的提升有所帮助，TextRCNN 的预测效果是最好的，准确率达到

79.31%。而基于预训练的深度学习模型中表现最好的是基于股吧文本的预训练 BERT 模型，准确率达到了 81.26%，F1 值为 0.8070，其次是 ENRIE 模型和已预训练好的 BERT 模型，Transformer 的效果最差。（2）基于股吧文本的预训练 BERT 模型构建的情绪指标与三种股指期货主力合约收益率均存在相关性。（3）使用 LSTM 预测股指期货价格时，情绪指标的加入能提升模型预测的准确率，均方误差降低了 17.85%，说明网络舆情能为投资者的决策提供一定的帮助。

关键字： 网络舆情　股指期货　预训练深度学习模型　BERT　LSTM

Abstract

Online public opinion is the sum of public remarks and emotional tendencies on a topic, event, product and brand on the Internet. With the popularity of the Internet and the rise of social media, online public opinion has become one of the important factors influencing social opinion and decision making. Online public opinion is closely related to the financial market: firstly, investors' decisions are often influenced by online public opinion, and some negative news and rumors may cause investors to sell in panic, thus negatively affecting stock prices. In addition, some information and predictions in online public opinion may affect the direction of stock prices, such as remarks of certain important figures and policy adjustments, which may become important bases for investors to make decisions. Therefore, the monitoring and analysis of online public opinion is important for predicting the trend of price changes in the financial market. In recent years, with the continuous development and popularity of social media, more and more investors have started to obtain market information through channels such as social media, which is more immediate and extensive compared with traditional media. Therefore, how to use big data analysis technology and machine learning methods to tap the value of information in online public opinion and predict the trend of financial market price changes has become a hot research topic in the financial field today.

Existing research on text sentiment recognition firstly includes traditional machine learning sentiment recognition models based on manually extracted features to train classifiers, which need to be determined by professional domain knowledge or experience, and thus rely more on manual experience and the quality of data pre-processing, and at the same time cannot make full use of the implicit information in large-scale corpus and cannot solve the ambiguity and complexity of language. Secondly, non-pre-trained deep learning emotion recognition models, which usually use neural networks to learn features from raw text data and train classifiers, feature automatic extraction of relevant features in text, avoiding the problems of feature engineering and manual feature extraction. However, since non-pre-trained deep learning models require more labeled data for training and are susceptible to noise in the dataset. Finally, there is a pre-trained deep learning emotion recognition model based on pre-training, which is characterized by the ability to pre-train using a large amount of unlabeled data, resulting in better text representation with better transferability and generalization performance. However, pre-trained models require larger computational resources and datasets, and more domain expertise for tuning and optimization.

TextRCNN, TextRNN_Att, BERT, BERT_CNN, BERT_DPCNN, BERT_RNN, ENRIE and BERT_RCNNN are selected for training tests, after which their F1 scores and accuracies are reported, specifically comparing BERT with those based on The results of BERT after fi-

nancial text pre-training are specifically compared. After selecting the models with the best prediction results, the sentiment indicators are constructed and their correlations with the returns of the three main stock index futures contracts, IF00, IC00 and IH00, are calculated. Finally, the LSTM model is used to forecast stock index futures prices, while comparing the effect of whether or not to include sentiment indicators on the prediction effectiveness of the model.

The conclusions drawn in this paper include: (1) the prediction effect of pre-trained deep learning sentiment recognition models based on pre-training is better than that of non-pre-trained deep learning sentiment recognition models. For the non-pre-trained deep learning model, appropriately increasing the complexity of the neural network is helpful for the improvement of the prediction effect, and the prediction effect of TextRCNN is the best, with an accuracy of 79.31%. The best performance among the pre-trained deep learning models is the pre-trained BERT model based on the stock bar text, with an accuracy of 81.26% and an F1 value of 0.8070, followed by the ENRIE model and the pre-trained BERT model, and the Transformer has the worst effect. (2) The sentiment indicators constructed by the pre-trained BERT model based on stock bar text are correlated with all three stock index futures main contracts. (3) When using LSTM to predict stock index futures prices, the addition of sentiment indicators can improve the accuracy of model prediction, and the mean square error is reduced by 17.85%, indicating that online public opinion can provide some help to investors' decision making.

KEY WORDS: Online Public Opinion　Stock Index Futures　Pre-trained Deep Learning Model　BERT　LSTM

目录

第一章　引言

第一节　研究背景

近五年以来，中国经济已经进入由“高速增长”转向“高质量发展”的新阶段，经济发展与结构优化问题正面临巨大考验，随着互联网的快速发展，大量信息资讯每天都在不断产生，每日产生的新闻数量已经超过了个体数年的阅读量。对于金融市场来说，财经新闻起到了晴雨表的作用。传统金融学框架中，投资者是理性的，做出的投资决策也是理性的，但是在现实中并非如此，投资者往往对未来收益的预期与经典金融学理论有着较大的偏差，投资者情绪是这种偏差的重要来源，随着行为金融学的发展，其理论对于这种异象问题有着较好的解释。通常投资者会带有一定的投机性情绪，比如对于市场层面、政策层面的态度以及对于企业财务数据预期的情绪，在整个周期中，投资者情绪上升往往伴随着投资者对利好信息的强化与对利空信息的弱化，市场整体的乐观化使得越来越多的投资者进入，股价进一步上升，伴随着对预期的进一步看好，投机者的需求进一步扩大，市场在高昂的情绪中逐步见顶；随着投资者和市场回归理性，利空消息被强化，利好消息被弱化，市场在投资者情绪的下降逐步见底。近年来，许多国内外研究表明舆情对于市场整体情绪有着较大的影响，张祚超等（2021）发现媒体新闻与收益波动率有显著的正相关关系。陆沁晔和陈昊（2021）的研究结果显示媒体报道导致投资者情绪产生积极或消极的变化，而投资者情绪对投资者决策产生直接的影响，即媒体报道倾向性引发股价波动。

财经新闻通过媒体等各类平台向社会发布，包括公司财报是否达到预期、公司高管任免、政策类新闻、国内外形势等。而这些信息资讯会成为分析师、机构与个人投资者投资决策的重要依据，通过对于信息的解读，可以在一定程度上识别投资风险，发现投资机会，对于整个金融市场的影响是持续性的。除了新闻类资讯，金融机构分析师给出的研究报告也是一种重要的参考，例如朝阳永续根据卖方分析师群体对上市公司业绩预测的共识与分歧建立的上市公司盈利预测数据库，包括国内 A 股卖方原始预测数据与指数和分行业的一致预期数据，这是对于金融市场预测的重要依据之一。相比于市场情绪，网络舆情的传播性更广，并且更容易被意见领袖传播扩散，进一步放大强化其作用，舆情推动着投资者做出一些非理性的行为，进一步对金融市场产生影响。

从另一方面看，舆情数据作为另类数据里重要的一种，目前广泛应用于公募、私募投研团队与卖方金工团队。另类数据，泛指区别于传统金融数据的有价值的信息和数据，有着体量大、流动速度大，种类多的特点。IDC 公司的报告显示，2018 年全球有 33ZB 的数据，预计 2025 年这个数字会增长到 175ZB。近年来随着国内量化投资的高速发展，因子拥挤与策略同质化的现象日益明显，另类数据因其获取与处理计算方面的门槛的存在，可以为投资提供相对差异化的 Alpha 的来源，可以有效减小因子拥挤与策略同质的问题，因此在量化投资中的应用逐渐广泛，其包含的基本面信息，对股票收益的预测能

力的有效性也得到了充分的证明。以中信证券发布的研究报告“基本面量化专题：另类数据 + 行业逻辑，赋能基本面量化”为例，对于汽车行业的研究使用的另类数据有汽车行业政策热度、明星机构统计、汽车重点公司研究热度等，特别的使用了 Bilibili 消费者心智预览、百度指数概览、品牌用户阅览数量和车型评论数量与文本这些舆情数据应用于行业逻辑，挑选出可以创造超额收益的标的，将相应的选股逻辑应用于量化策略中。但是，目前在如何使用另类数据这种新型数据上会面临一些障碍与风险，并非所有另类数据在金融市场都有获取潜在 Alpha 的能力，首先需要判断数据是否存在价值；与此同时，对于另类数据的收集与清晰同样是使用时的难点，同样还存在潜在的隐私问题、监管问题以及无效应用等问题。

第二节　研究意义

本文具有一定的理论与现实意义，首先是将组成另类数据重要部分的网络舆情数据与金融市场有效结合，能够帮助投资者更有效的识别市场情绪，其次通过 NLP 模型将网络舆情量化打分，建立舆情与金融市场的联系。现实意义方面，随着股指期货限制逐渐放开，国内公募、私募对冲产品空单持仓在三大股指持仓中占较大比重，股指期货作为股市提供对冲的重要工具，以沪深 300 股指期货为例，仅 2021 年交易额就达到了 163 万亿元，作为金融市场的重要组成部分，网络舆情对于期货市场尤其是股指期货市场收益率如何影响，具有非常重要的研究意义。

第三节　论文创新点

目前国内外对于网络新闻舆情对金融市场的影响的相关研究相对成熟，但是存在以下三方面的不足：

（1）文献主要集中在股市、商品期货和国外股指期货的舆情资讯，针对国内股指期货的相关研究较少。受限于我国与世界其他国家资本市场存在的差异，应当加强国内网络新闻舆情对股指期货收益率的影响的研究。本文的主要研究对象即为我国网络新闻舆情，同国际市场的网络新闻舆情数据形成比较，研究跨市场的影响，相关研究结论能够更好地服务中国。（2）现有的相关研究大多使用传统的机器学习方法，如随机森林、支持向量机、神经网络等。缺乏对 BERT_RCNN、ENRIE 等最新前沿方法的实际应用和方法综合。本文综合考虑 11 种深度学习模型，对不同深度学习方法进行综合，选取最合适的模型进行预测，进行方法上的创新，具有更好的预测效果。（3）网络新闻舆情对股指期货收益率的影响受到很多文本情境因素的影响，以往的研究主要考虑政治、经济、社会等因素，本文专门采用金融文本的语境进行训练，提高了对于金融文本识别的准确性。

综上所述，本文将网络舆情对于投资者情绪的影响，进而对股指期货的收益率的影

响深度分析，并进一步对比投资者情绪的高低时期现货市场的波动差异。同时针对已经预训练的 BERT 模型进一步进行金融文本训练，提升了模型对金融文本的情感识别的准确性。最后将情绪指标纳入深度学习预测模型，提升了对股指期货价格的预测能力。

第四节　结构安排

本文选取了 2018 年 1 月 3 日至 2023 年 1 月 20 日东方财富网股指期货股吧的帖子标题、发帖时间作为网络舆情情感挖掘文本，同时选取这段时间的 IF00、IH00 与 IC00 三种股指期货主力合约的日频数据作为研究对象，数据来源为天软（Tinysoft）。具体结构安排如下。

第一部分是引言，介绍了本文的选题背景，投资者情绪对投资者决策起到至关重要的作用，而随着占据另类数据中较大比例的舆情数据的日益增长，投资者情绪也随之受到较大影响。

第二部分是国内外研究现状与发展趋势，对于投资者情绪与其度量方法研究进行综述，其次对网络舆情如何作用于投资者情绪问题进行综述，进一步对投资者情绪如何影响金融市场进行综述，最后对文本情感挖掘分析方法进行综述，该部分对本文后面章节的写作提供了较大启发与参考。

第三部分是文本情绪识别的模型研究，重点介绍了常见的文本情感识别提取模型，包括。该部分对于第四部分实证研究提供了理论基础，并基于此进行后续的优化。

第四部分是网络舆情对股指期货收益率影响实证分析，首先通过爬虫对东方财富股指期货股吧 2018 年 1 月至 2023 年 1 月的帖子标题进行获取，标注了 5863 条数据情感后将数据切分映射至文本情绪识别模型中，并且进一步调整模型，选择出对情绪识别准确率最高的模型对完整数据集进行情绪标注。同时构建情绪指标对应至每个交易日，验证各股指期货主力合约与其相关性，最后使用加入情绪指标的 LSTM 模型对股指期货价格进行预测。

第五部分是结论与展望，基于第四部分得出的结论得出了网络舆情对股指期货收益率的影响，并为投资者与监管机构等提出相关建议，最后总结了未来研究展望。

最后是参考文献与附录，其中附录包括了 BERT 模型训练与基于情绪指标的预测股指期货价格的 LSTM 模型代码。

本文的研究方法包括：文献研究法；基于金融文本训练的 BERT 情绪识别方法和基于情绪指标的 LSTM 预测方法。

第二章　国内外研究现状与发展趋势

第一节　投资者情绪与其度量方法

在传统金融学的框架下，股票价值主要是由其上市公司的基本面决定的，但是通常在短期内企业的基本面并不会发生明显的变化，而股价却经常发生短期剧烈波动的现象，这是因为投资者情绪发挥了主要作用。

Lee 等（1991）使用了封闭式基金的折现率作为投资者情绪的代理变量，发现该代理变量有效，基金的风险直接受到投资者情绪影响，从而影响资产组合的价格。Brown 和 Cliff（2006）发现了投资者的预期导致了市场的波动，投资者情绪本质上反映了市场参与者的期望。Baker 和 Stein（2003）通过换手率作为反映流动性的指标代表投资者情绪，发现在流动性强的市场中，价格主要由非理性的投资者主导，高流动性反映了投资者的积极情绪。Peng 和 Xiong（2005）提出投资者情绪是一种稀缺的认知资源，并且对投资者关注点进行建模，发现有限的投资者关注会导致存在类别学习的现象。Baker 和 Wurgler（2006）通过构建了 IPO 数量、股权、换手率、首日平均收益率和股息溢价等复合指标度量投资者情绪，发现投资者情绪像跷跷板一样决定了投资者如何选择投资标的。Aboody 等（2018）对于投资者情绪使用了隔夜收益率来衡量，投资者情绪对市场短期收益率有着显著的正向作用，同时低隔夜收益率的股票长期表现更好。这种以代理变量形式表示的投资者情绪存在着一定的局限性，同时仅仅采用传统金融学的计量方法研究在现在的大数据背景下也略显单薄，随着另类数据的发展，网络新闻舆情被发现可以通过影响投资者情绪的方式作用于金融市场。

第二节　网络舆情如何作用于投资者情绪

Antweile 与 Frank（2004）采用金融网站上的文本数据，通过统计学算法分类后形成-1,0,1 三类反映消息的情绪，研究情绪对金融市场价格的影响。Zimbra 等（2009）发现企业论坛中即使发布无关股市的信息，但只要与公司层面相关也会对投资者情绪造成影响，从而间接作用于企业股价。Bollen 等（2011）通过 twitter 中提取的推文文本衡量公众情绪状态，发现投资者情绪提高会给股票带来高波动，同时社会传染效应会放大谣言与误导信息导致市场波动幅度增加。Yang 和 Yan（2011）发现新闻舆情导致的高情绪在超过某个临界值后会带来负的超额收益，低于该临界值则超额收益为正。朱南丽等（2015）使用新浪博客与新浪微博数据量作为投资者情绪的代理变量，证明了其有效性以及其更好地反映了个人投资者的情绪。李思龙等（2018）采用了东方财富股吧论坛帖子考察投资者情绪，发现投资者的互动可以增加企业股东数量并且提高流动性，同时降低信息不对称性。

第三节　投资者情绪如何影响金融市场

张强等（2007）的研究表明，相比于几乎不造成影响的个人投资者，机构投资者情绪是影响中国股市的系统因素。Lin 等（2018）研究投资者情绪与美国股指期货市场以及其对应的现货标的价格的关系，发现投资者情绪与期货市场表现呈现明显的负相关关系。Cedric 等（2019）通过研究投资者关注度对投资者情绪的影响，发现其因果关系，并且投资者情绪对于大盘股而言较为短暂，对于小盘股而言情绪效应较为持久。龙文等（2019）考察财经话题与行业收益率的影响，发现财经话题分布变化会导致行业热点的转移，投资者情绪会从行业层面影响金融市场。孙明宏（2021）通过对于金融文本的学习提取出金融舆情指标，发现其稳定性以及对个股的较强预测能力。赖星武（2021）使用网络舆情文本构建综合情绪指标，发现指标与黄金期货价格存在显著的非相关性。目前，针对投资者情绪对股市、商品期货以及国外股指期货研究较多，而国内股指期货与网络舆情的相关研究较少。

第四节　文本情感挖掘分析方法

Kim（2014）将卷积神经网络（CNN）用于句子分类任务，包括使用多个卷积过滤器和最大集合层来从输入的句子表示中提取局部特征，发现 CNN 在句子分类任务中表现优于传统的机器学习算法，并且强调了这种方法进一步发展和改进的潜力。Lai 等（2015）结合了递归神经网络（RNN）和卷积神经网络的优势来解决文本分类的任务，提出了一个新的架构，通过使用卷积层，既考虑了文本数据的顺序性，又考虑了局部特征，在文本分类任务上的表现优于传统的 RNN 和 CNN。Liu 等（2016）表示，尽管许多研究已经证明 RNN 在文本分类任务中具有较高的效率，但单一的 RNN 模型在处理多任务分类问题时存在缺陷，因此提出了一种基于 RNN 的多任务学习方法，通过结合两者优势以提高文本分类的效果。Peng 等（2016）提出使用基于注意力的双向长短期记忆（Bi-LSTM）网络来处理文本数据中实体之间关系的表示和分类的问题，该架构通过注意力机制能够高效地从过去和未来的输入中捕捉上下文信息，为关系分类问题作出了较大贡献。Johnson 和 Tong（2017）提出了一个深度金字塔 CNN 架构，它由多个卷积层和池化层组成，旨在捕捉文本的局部和整体特征，其效果优于传统的浅层 CNN，为文本分类任务的深度 CNN 架构的设计提供了深入的见解。Vaswani 等（2017）首次提出了一种全新的不依赖于序列信息的注意力机制：Transformer，大大简化了网络的架构，同时性能也表现十分优异，推动了注意力机制在自然语言处理领域的广泛应用，并对深度学习领域产生了重要影响。Hu 等（2017）使用 HAN 模型，通过分层自注意力机制来处理文本中的不同级别信息，以捕捉文本情绪的含义，分层注意力包括词级别的注意力和句子级别的注意力，分别计算词语和句子对情绪预测的重要性，发现其能够更加有效地识别文本情绪。Devlin 等（2018）提出了一种使用深度双向变换器（BERT）进行语言

理解的新的预训练方法。与传统的只用文本来预测下一个词的预训练模型不同，BERT在其预训练过程中同时考虑到了上下文，该模型在预训练语言表示任务中表现超过以往最先进的模型，对 NLP 社区产生了重大影响，并激发了许多后续研究和实际应用。Sun等（2019）提出了名为 ERNIE 的模型，在其预训练过程中同时利用了非结构化和结构化的知识，这使得它能够更好地捕捉词语之间的关系和文本的意义，通过增加额外的预训练任务来实现，这些任务涉及外部知识源的整合，如实体级信息、部分语音标签和命名实体识别，旨在提高预先训练的语境表征的质量。它能够有效地利用知识整合来增强语言表示能力。

第三章　文本情绪识别的模型研究

由于金融文本数量较多，文本间存在较多上下文之间的非线性特征，其专有名词在不同语境下的意义可能截然相反，因此传统的机器学习算法虽然相对简单，易于理解和实现，可以处理一些高维度的特征，但是却无法很好地处理文本语义信息，不能自动学习语义表示，同时对于高维度、稀疏性和非线性的特征表示效果较差。模型的泛化能力和稳健性有限，很容易受到数据质量、特征选择等因素的影响。因此并不适合使用传统机器学习方法对金融文本进行研究，而基于深度学习的模型（如 TextCNN、BERT 等）在语义表示和模型性能等方面有着更好的表现，如 TextCNN 可以更好地捕捉文本中的局部特征，并且有着良好的可解释性与可扩展性，而 BERT 具有强大的学习能力，能够学习到更丰富的语义信息和上下文关系，同时具有强大的学习能力，能够学习到更丰富的语义信息和上下文关系，在各种文本分类任务上都有很好的表现，最关键的是 BERT 预训练的过程是无监督的，可以使用大量的未标记文本数据来预训练模型，从而避免了对大量标注数据的需求，此外也可以利用 Transformer 结构的优势，在处理长文本时能够更好地捕捉全局的上下文关系。

考虑到不同文本分类模型的训练速度和分类精度的优劣差异，本文选取了非预训练的深度学习模型以及基于预训练的深度学习模型：TextCNN、TextRNN、TextRCNN、TextRNN_Att 以及 Transformer、BERT、BERT_CNN、BERT_DPCNN、BERT_RNN、BERT_RCNN 和 ENRIE 这十一个模型用于研究，各模型的具体原理和作用如下。

第一节　非预训练深度学习情绪识别模型

一、 TextCNN 模型

TextCNN（Convolutional Neural Networks for Text Classification）是 Kim 等在 2014 年提出的一种基于卷积神经网络的文本分类模型。它将卷积神经网络（CNN）的应用于文本和情感识别，通过将文本转换为卷积神经网络可以处理的形式，然后使用卷积神经网络提取文本特征，进而实现文本分类。

主要由三个部分组成：输入层、卷积层和池化层。输入层将文本转换为卷积神经网络可以处理的形式，通常是一个矩阵，其中每一行代表一个单词，每一列代表一个特征，比如词性、词频等。卷积层使用卷积核对输入层的输入进行卷积操作，从而提取文本特征。池化层使用池化操作，将卷积层的输出进行降维，从而减少计算量。TextCNN 的特点是可以通过卷积窗口的大小从文本中提取不同 N-gram（N 小于等于窗口大小）的特征，并将这些特征提取出来做拼接，并转换成合适的输入特征，输入到全连接层，训练得出最终的类别结果。

TextCNN 可以通过增加更多的层来提取更丰富的特征，而多层网络可以更好地抽取文本特征，从而使模型表现更加出色。TextCNN 网络结构简单，参数较少，具有较快

的训练速度和收敛速度，但是只能处理短文本。

TextCNN 模型的第一层是输入层，表示一个 $n \times k$ 的矩阵，其中 n 为句中词汇数，k 为对应词向量的维度，每行就是一个单词对应的 k 维词向量，本文使用 $x_i \in \mathbb{R}^k$ 表示句中第 i 个单词的 k 维词嵌入。

第二层为卷积层，输入矩阵进行卷积操作，产生一个特征 c_i，即:

$$c_i = f\left(w \cdot x_{i:i+h-1} + b\right) \tag{3.1}$$

其中 $x_{i:i+h-1}$ 表示输入矩阵的第 i 行到第 $i+h-1$ 行组成的窗口，h 为窗口的词汇数，w 为权重矩阵，b 为偏置参数，f 为函数，一步步进行点积运算，在 $x_{1:h}$ 卷积得到 c_1，在 $x_{2:h}$ 卷积得到 c_2，通过提取特征向量，从而构建卷积层输出，可得:

$$c = [c_1, c_2, \ldots, c_{n-h+1}] \tag{3.2}$$

最后在池化层对各窗口生成的特征向量笃选最大的值并拼接，即将不同长度的句子通过最大池化得到定长的向量。

$$c_{\max} = \max(c) \tag{3.3}$$

二、 TextRNN 模型

TextRNN（Recurrent Neural Network for Text Classification）是一种基于循环神经网络（RNN）的深度学习模型，在 2016 年由 Liu 等人提出 [21]，可以自动学习文本中的语义特征，有效地处理不同长度的文本序列，通常应用于文本分类、情感分析、机器翻译、自动问答和文本生成等领域。

TextRNN 主要由 RNN（循环神经网络）和全连接层构成，RNN 及其变体包含 LSTM（长短时记忆网络）和 GRU（门控循环神经网络）等。在训练期间，它逐步地处理文本中的每个字符，并在处理过程中从前一步学习特征。当它处理完文本中的所有字符之后，它会根据从 RNN 层输出的向量和权重向量构建一个全连接层，以得出最终的预测结果。

TextRNN 可以有效地捕获文本中关键词和句子层次结构的信息和特征。首先，RNN 可以通过循环操作，逐步提取文本中的每个单词，并将其转换为一个特征向量，以便有效地提取文本的特征。其次，RNN 的变体 LSTM 和 GRU 可以更好地捕获文本的语义信息，因为它们可以更好地捕捉文本中的长期依赖性。最后，TextRNN 通过全连接层可以实现特征的混合，得到更准确的结果。此外，TextRNN 也被广泛应用于情感分析，TextRNN 首先从 RNN 层获取单词的特征向量，然后输入全连接层，这里会使用不同的激活函数得到情感相关的预测结果。

TextRNN 模型的输入端同样为词向量权重矩阵，并且最初设定步长参数，根据设定模型训练步长统一输入文本长度。Bi-GRU 层根据当前步长的隐层输出 x_t 的更新门和重

置门更新上一时刻的隐层输出 h_{t-1}，更新门和重置门公式如下:

$$\begin{aligned} z_t &= \sigma\left(W_z x_t + U_z h_{t-1}\right) \\ r_t &= \sigma\left(W_t x_t + U_r h_{t-1}\right) \end{aligned} \tag{3.4}$$

最后的输出层为常见的 $softmax$ 函数。模型进行训练时，为防梯度爆炸导致模型不收敛，一般采取梯度裁剪法：

$$t = \begin{cases} t \times \frac{\text{clip\_norm}}{\|t\|_2}, & \|t\|_2 \geq \text{ clip\_norm} \\ t, & \text{otherwise} \end{cases} \tag{3.5}$$

三、 TextRCNN 模型

TextRCNN（Recurrent Convolutional Neural Networks for Text Classification）在 2015 年由 Lai 等人提出，是一种基于循环神经网络（RNN）的文本分类模型，其中包含三部分：一个循环神经网络层（RNN），一个多层感知机层（MLP），以及一个最大池化层。TextRCNN 使用序列模型来提取文本特征；多层感知机层用于组合 RNN 层提取的特征；最大池化层提取文本特征并将其映射到固定长度的向量，用于最后的分类。

TextRCNN 模型的输入是样本中每个词的词向量表示，模型将会进行编码，将每个词编码成一个词向量，然后将这些词向量传入到 RNN 层，用于捕捉句子特征。RNN 层采用双向循环神经网络实现，其中，一个向前的 RNN 负责从头到尾捕捉词级特征，另一个向后的 RNN 则从尾到头捕捉反向信息。RNN 层输出是双向特征，也就是每个单词的前向向量和反向向量。多层感知机层则将 RNN 输出的句子级特征组合起来，得到一个固定长度的向量。最大池化层则会从这个向量中提取最重要的特征。最终分类的结果应用到最后的池化结果上，从而得到文本分类的结果。TextRCNN 模型可以有效地处理文本中的噪声，并能有效地处理文本中的多义性。

TextRCNN 模型使用双向网络处理输入的向量，在各个时间步把循环网络的输出和对应词向量拼接，此时语义向量可以识别上下文特征，最后选择最大池化层的特征作为输出特征。对于文本 $x_{1:n}$，若 $c_l(x_i)$ 为第 i 个词的左语义向量，$c_r(x_i)$ 为第 i 个词的右语义向量，$e(x_i)$ 为第 i 个词的语向量，则有：

$$\begin{aligned} c_l\left(x_i\right) &= f\left(W^l \cdot c_l\left(x_{i-1}\right) + W^{sl} \cdot \mathrm{e}\left(x_{i-1}\right)\right) \\ c_r\left(x_i\right) &= f\left(W^r \cdot c_r\left(x_{i+1}\right) + W^{sr} \cdot \mathrm{e}\left(x_{i+1}\right)\right) \end{aligned} \tag{3.6}$$

拼接词向量和对应的左右语义向量，可得：

$$t_i = \tanh\left(W\left[c_l\left(x_i\right); \mathrm{e}\left(x_i\right); c_r\left(x_i\right)\right] + \mathrm{b}\right) \tag{3.7}$$

最后对于所有语义向量输出最大池化的特征，即:

$$t = \max_{i=1}^{n}\left(t_i\right) \tag{3.8}$$

四、 TextRNN_Att 模型

TextRNN_Att（Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification）模型结合了 RNN 和 Attention 机制，通常包含三个部分：LSTM 网络、Attention 机制和 Softmax 层。RNN 层用于捕捉文本中的上下文信息，Attention 层用于分析文本中每个词的重要性，Softmax 层用于将 RNN 层和 Attention 层的输出转换为文本分类的结果，得到预测的最终结果。

模型可以捕捉句子的上下文相关的语义特征，同时又加入了 Attention 机制，使模型能够更好地对文档中的重要信息进行重视。LSTM 网络可以捕捉句子的上下文相关特征，比只使用 Attention 机制的模型，可以得到更加准确的结果。

TextRNN_Att 模型实现以可扩展性、低脆度及可靠性为特点的高效文本分析预测。但是其计算量较大，因为必须计算每个词向量以及每个隐藏层单元，而且 LSTM 网络和 Attention 机制都比较复杂，调参需要许多实验，从而增加了实现时间，同时模型训练时间更长。

其中 Attention 分类模型流程如下，对于文本 $x_{1:n}$，使用 LSTM 网络得到第 i 个时间步的隐藏状态：

$$h_i = bi\_LSTM\left(x_i\right) \tag{3.9}$$

进一步地

$$\begin{aligned} u_i &= \tanh\left(Wh_i + b\right), \\ \alpha_i &= \frac{\exp\left(u_i^T u_w\right)}{\sum_i \exp\left(u_i^T u_w\right)} \\ s &= \sum_i \alpha_i h_i \end{aligned} \tag{3.10}$$

其中 u_w 是模型训练时更新的参数，α_i 是注意力权重。

第二节　基于预训练的深度学习情绪识别模型

一、 Transformer 模型

Transformer 模型原理基于注意力机制，由 Google 研究人员 Vaswani 等人在 2017 年提出。Transformer 模型使用一种多头注意力的技术，这种机制可以在序列中计算每个单词的重要性，并在序列之间计算相似性。模型可以计算给定单词的注意力分布，以及输入和输出之间的注意力分布，将不同位置的单词拼凑在一起，形成一句完整的话，以及识别句子中有用的信息。Transformer 模型还使用了位置编码技术、层级编码技术和残差连接技术等，使得模型更好地理解文本中的语义关系，并且可以捕捉文本中的更复杂

的关系。Transformer 的核心是将 NLP 问题解决问题的多层结构编码为抽象变换，而不是按照标准的语言模型。Transformer 模型的原理结构如图3.1所示：

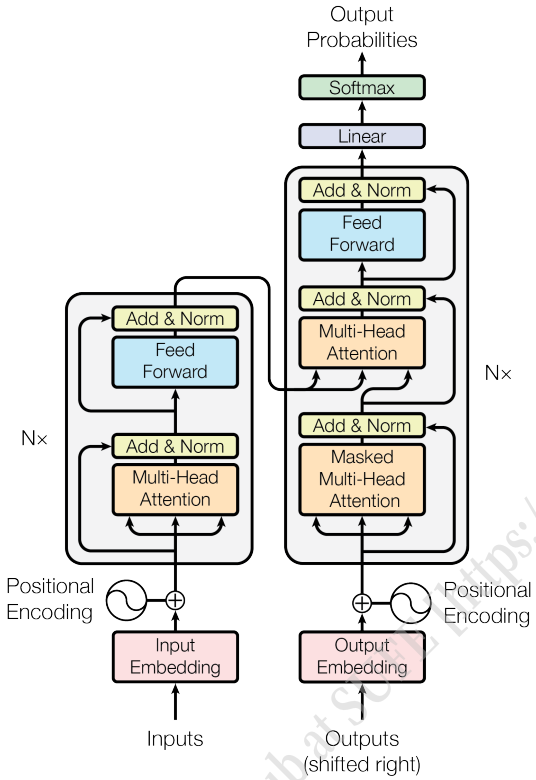

图 **3.1**　**Transformer** 框架结构

从图中可以看出，Transformer 由编码器（Encoder）与解码器（Decoder）组成，每个编码器由 1 个位置编码层与 N 个编码层（Encoder Layer）组成，每个解码器由 1 个位置编码层与 N 个解码层（Decoder Layer）以及 1 个以 $softmax$ 为激活函数的全连接层组成，第 t 个编码层的输入是 t-1 个编码层的输出，解码层同理。最终编码器的输出将作为解码器中某个模块的输入，解码器的输出即整个模型的输出是 $softmax$ 归一化后序列类别分布，通常维度是 [Batch Size，序列长度，类别数量]。

输入传入 Embedding 层后完成 Word Embedding 后，首先经过位置编码（Positional Encoding）以获取词向量在句子中的位置信息。

$$emb_{\text{out}} = emb_{\text{in}} + PE_{\text{in}} \tag{3.11}$$

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{3.12}$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{3.13}$$

其中 emb_{in} 和 emb_{out} 分别是位置编码的输入和输出，pos 是词在句子中的位置，i 表示词向量的第 i 个维度，d_{model} 则表示模型的维度。之所以将 PE 设置为三角函数，是因为

可以利用其性质得到所有位置之间的相对位置信息：

$$\begin{aligned} PE(M+N,2i) &= PE(M,2i) \times PE(N,2i+1) + PE(M,2i+1) \times PE(N,2i) \\ PE(M+N,2i+1) &= PE(M,2i+1) \times PE(N,2i+1) + PE(M,2i) \times PE(N,2i) \end{aligned} \tag{3.14}$$

所以 $PE(M+N)$ 可由 $PE(M)$ 与 $PE(N)$ 计算得到，也就是说各个位置间可以相互计算得到。

位置编码完成后是编码层，包括多头注意力、残差 &Layer Normalization 与前馈神经网络三个部分组成。对于多头注意力，通常采用缩放点乘注意力（Scaled Dot-Product Attention）算法计算：

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}V\right) \tag{3.15}$$

$$\begin{aligned} Q &= W_q \times X + b_q \\ K &= W_k \times X + b_k \\ V &= W_v \times X + b_v \end{aligned} \tag{3.16}$$

其中 d_k 指 K 向量的维度，Q 代表 Query 向量，K 代表 Key 向量，V 代表 Value 向量。在“编码器”中，Q,K,V 都是由输入的序列向量得到。$W_q, b_q, W_k, b_k, W_v, b_v$ 是模型需训练的三套不同的线性变化参数。

以上的注意力层可称为单头注意力层，而多头注意力 Multi-Head Attention 是指用不同的 $W_q^i, b_q^i, W_k^i, b_k^i, W_v^i, b_v^i$ 的多计算几次单头注意力层的输出之后再全部拼接起来:

$$head_i = Attention\left(Q_i, K_i, V_i\right) \tag{3.17}$$

$$MultiHead(Q,K,V) = concate\left(head_1, head_2, \ldots, head_h\right) W^c \tag{3.18}$$

其中 W^c 是模型需训练的线性变化矩阵，维度为单头注意力层输出向量的维度 ×head 数量，多头注意力层输入向量的维度，即输入向量 X 的维度。它的作用就是将经过多头注意力操作的向量维度再调整至输入时候的维度。

多头注意力后是残差，即在经神经网络多层传递后加上最初的向量，这样的好处是根据链式求导法则，不管网络多深，梯度上都会至少有 1，不会为 0 造成梯度消失。Layer Normalization 是数据规范化的操作，公式如下：

$$\widehat{\boldsymbol{a}}^l = \frac{\boldsymbol{a}^l - \mu^l}{\sqrt{\left(\sigma^l\right)^2 + \varepsilon}} \tag{3.19}$$

其中 ε 是干扰因子，μ^l 代表第 l 个均值，σ^l 代表第 l 个均值，计算公式如下：

$$\begin{aligned}\mu^l &= \frac{1}{H}\sum_{i=1}^{H} a_i^l \\ \sigma^l &= \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2}\end{aligned} \tag{3.20}$$

经过编码器输出后则是解码器。相较于编码器多了遮盖的多头注意力层与交互注意力层，在训练时解码器的输入是要预测的序列。对于遮盖的多头注意力层，遮盖的意义是为了将未来信息掩盖住，使得训练出来的模型更准确。例如输入“我爱新中国”，当轮到要预测“中”时，模型获得的信息应该是“我爱新”这三个字。但是 Transformer 的 Attention 层做计算时是一整个序列张量输入进行计算，所以如果不加处理，预测“中”这个字时，模型获得的信息将会是“我爱新 * 国”这四个字。则“国”对于“中”来说显然属于未来信息。所以在训练时需要将未来信息都遮盖住。使模型在预测“中”时，获得的信息是“我爱新 **”。预测“新”时，获得的信息是“我爱 ***”。

解码器中的交互注意力层与编码器中的注意力层唯一区别在于，前者计算 Query 向量的输入是编码器的输出。

二、 BERT 模型

BERT（Bidirectional Encoder Representation from Transformers）模型属于双向语言模型，在 2018 年 10 月首次被 Google AI 研究院提出 [26]，它从句子的两端同时学习语义和语法，根据上下文预测，这使得 BERT 模型更加灵活，可以应用于各个领域。例如自然语言生成（NLG），语音识别（ASR），自动文摘（AS）等。

BERT 模型基于一种叫做“转换器”的深度神经网络结构。该模型有俩大特点：双向性和自注意力。双向性使得模型能够从句子的右端和左端进行解读，而自注意力机制使得模型能够在复杂的句子中捕捉到非常有用的信息，比如短语和短语间的语法关系。BERT 模型比其他传统的模型更容易进行训练，不需要额外的数据预处理和特征工程步骤，大大简化了机器学习流程，可以使用受限的计算资源来运行，减少设备成本。

BERT 模型主要由单词向量、句子向量和位置向量组成，其中词向量用于判断句首，以便开展后续文本分类，句向量用于区分上下两句话。模型包括预训练和微调两个环节，如图3.2。其中预训练由两个自任务监督 MLM（掩蔽语言模型：Masked Language Modeling）和 NSP（下一个句子预测：Next Sentence Prediction）组成，产生一个联合训练的损失函数从而迭代更新整个模型中的参数。MLM 通过上下文预测输入文字中被随机掩盖的内容，NLP 判断下一句是否为该句的下文。

对于 BERT 的编码器而言，输入为一个句子，之后给句首添加 <CLS> 符号，两句句子中间添加 <SEP> 符号，句末也添加 <SEP> 符号，组成一组输入，紧接着在 Embedding

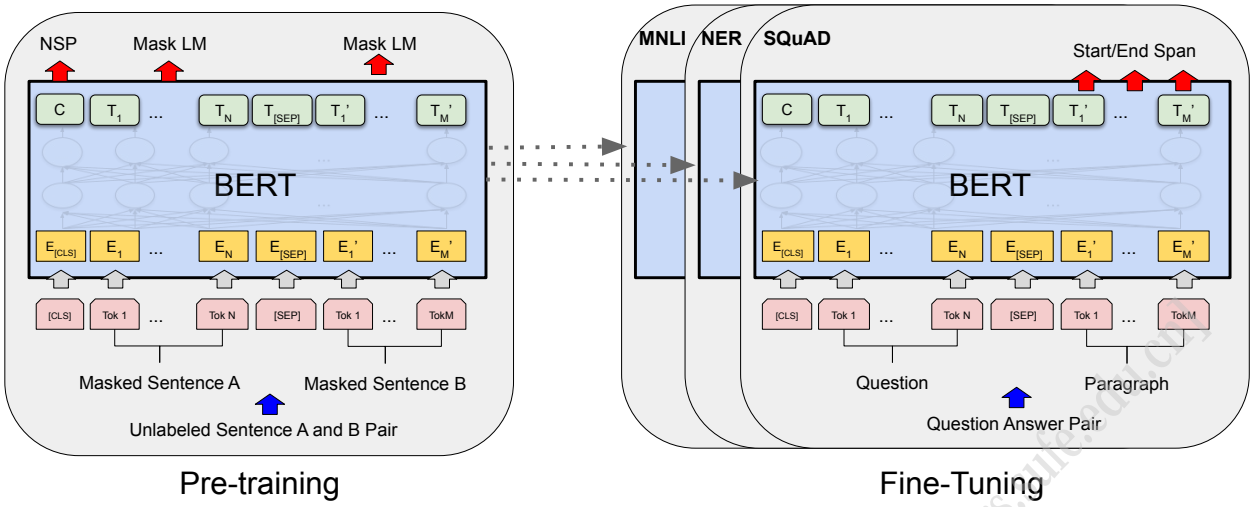

图 **3.2**　**BERT** 预训练与微调过程

层将输入组合成三种 Embedding 的相加，分别是 Token 级别、句子级别、位置级别的 Embedding，如图3.3。将输入传进若干层的 Transformer Encoder Layer，默认为 12。最后输出编码好的张量。

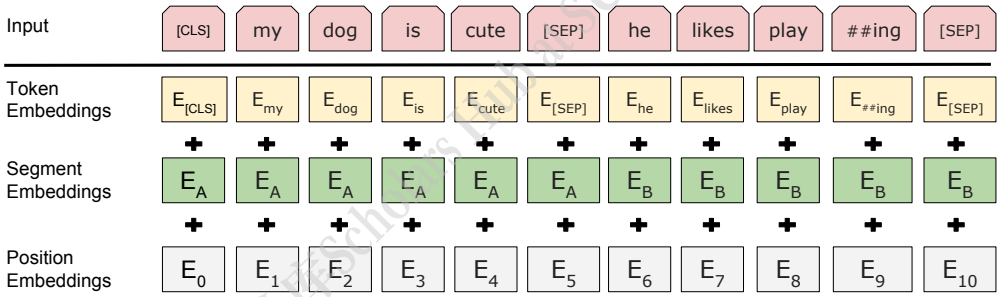

图 **3.3**　**BERT** 输入 **embedding** 层的过程

MLM 任务的输入为 Bert 编码器层的输出，需要预测的词元位置，将其传入一个中间层，包括 ReLu 层、Layer Normalization 层和一个全连接层，最后输出一个序列类别分布张量。特别地，MLM 任务包括一个采样的子任务：

- 对每一对句子中随机选择 15% 位置的词语进行遮盖动作
- 遮盖时 80% 替换为“<mask>”，10% 替换为随机词元，10% 不变。过程中 <CLS> 与 <SEP> 不会被替换

NSP 任务的输入为 Bert 编码器层的输出 <CLS> 位置的张量（也就是序列中的首位）。中间层与 MLM 一样是一个 MLP 的结构，默认是一个输出维度为 2 的，激活函数为 $softmax$ 的全连接层。对于 NSP 任务而言，其概率表达式为：

$$P = softmax(CW^T) \tag{3.21}$$

其中 C 是 BERT 模型 $softmax$ 函数输出的句首对应的向量，W 是连接层内可学习的权

值矩阵。

最后输出类别分布张量，NSP 任务采样时，50% 的概率将第二句句子随机替换为段落中的任意一句句子。

BERT 模型完整训练过程为：

- 通过文本整理出索引化的 Tokens 和 Segments、MLM 任务要预测的位置和真实标注与 NSP 任务的真实标注
- 将索引化的 Tokens 和 Segments 输入 Bert Encoder 得到编码后的向量
- 输入 MLM 网络得到预测值，并与真实标注计算交叉熵损失函数
- 从编码后的向量中取出 <CLS> 对应位置的张量输入 NSP 网络得到值与真实标注计算交叉熵损失函数
- 将 MLM 的 loss 与 NSP 的 loss 相加得到总的 loss，并反向传播更新所有模型参数

其中交叉熵损失函数（Cross Entropy Loss）表示预测数据概率分布与真实数据概率分布的差异，值越小则说明预测效果越好，其计算公式为

$$H(P,Q)=H(P)+D_{KL}(P\|Q)=-\sum_{i=1}^{n}P\left(x_i\right)\log Q\left(x_i\right) \tag{3.22}$$

$p(x)$ 表示样本的真实分布，$q(x)$ 表示模型所预测的分布。

最后是模型的微调，主要是指通过预训练得到的 BERT Encoder 网络接上各种各样的下游网络进行不同的任务。在 Google AI 研究院首次提出时，被指出其主要可以应用于 4 大类任务：

- 句子对分类，将经过 Encoder 层编码后的 <CLS> 对应位置的向量输入进一个多分类的 MLP 网络中即可
- 单句分类，同上
- 根据问题得到答案，输入是一个问题与一段描述组成的句子对。将经过 Encoder 层编码后的每个词元对应位置的向量输入进 3 分类的 MLP 网络，而类别分别是 Start（答案的首位），End（答案的末尾），Span（其他位置）
- 命名实体识别，将经过 Encoder 层编码后每个词元对应位置的向量输入进一个多分类的 MLP 网络中即可

实际上，在实际应用中除原论文中给出的四大类任务，也可以结合实际场景设计更多的微调任务。本文后面涉及到的主要是 BERT 的单句分类任务。

三、 BERT_CNN 模型

BERT_CNN 模型结合了 BERT 和卷积神经网络（CNN）的优势。BERT 模型是一种双向语言模型，它可以捕捉句子中的上下文信息，而 CNN 模型可以捕捉句子中的空间特征。Bert_CNN 模型由三个部分组成：BERT 预训练阶段，TextCNN 阶段和 BERT +

TextCNN 阶段。首先，在 BERT 预训练阶段，通过 BERT 的预训练模型构建双向语言模型，以学习文本中的长期依赖关系。对每一句话，BERT 预训练模型都会生成一个对应的向量，用于将句子内容映射到一个固定维度的语义空间中。其次，在 TextCNN 阶段，BERT 预训练模型中产生的句子向量将作为 TextCNN 模型的输入，通过模型运行，TextCNN 模型使用深层卷积神经网络来捕获文本中复杂的结构，层级表示，对文本进行分类。最后在 BERT + TextCNN 阶段，将 BERT 和 TextCNN 的输出结果进行融合，使用 $softmax$ 激活函数最终得出文本的类别结果。

利用 BERT 构建双向语言模型学习文本中的长期依赖关系，其中 BERT 在模型融合中可以将语义知识和长期信息融入到 TextCNN 模型中，而 TextCNN 模型则能够捕获文本的结构和表示，从宏观上提高文本分类的准确率。BERT_CNN 模型还可以更好地处理长句子，因为它结合了 BERT 和 CNN 的优势。

四、 BERT_DPCNN 模型

BERT_DPCNN（Deep Pyramid Convolutional Neural Networks for Text Categorization）模型结合了 BERT 模型和双向穿越卷积神经网络（DPCNN）模型的优势，最早由 Johnson 在 2017 年提出 [23]。BERT_DPCNN 模型的架构如下：首先，使用 BERT 模型对输入文本进行编码，以捕捉句子中的上下文信息；并且将 BERT 编码的输出作为 DPCNN 的输入，使用 DPCNN 模型捕捉文本中的特征；最后，将 DPCNN 模型的输出作为分类器的输入，使用分类器对文本进行分类。

模型的输入是一个句子，会将句子中的每个词映射到一个向量，然后通过双向自注意力机制和双向穿越卷积神经网络来捕捉句子中的上下文信息和局部特征，最后将句子的向量表示作为输出。但是它的缺点在于它的训练时间较长，而且它的训练过程需要大量的数据。

五、 BERT_RNN 模型

BERT_RNN 模型由一个基础网络架构组成，该网络由 BERT（双向编码器基于注意力）和 RNN（循环神经网络）共同构成，BERT 用来捕捉耦合的语义和语法特征，而 RNN 用来捕捉上下文信息。综合这些特征，BERT_RNN 模型可以根据输入的文本建模其语义和语法特征。同时 BERT RNN 模型可以节省训练时间，因为它可以使用预训练的语言模型来提高模型的准确性。

除了可以帮助自然语言解读系统获取更多准确的文本信息，它还可以用来识别和提取文本中的意图，以及识别句子和单词的新词以及用户的语用意图。此外，BERT RNN 模型还可以用于计算机视觉，如图像分类和目标检测，以及计算机音频等领域。

六、 BERT_RCNN 模型

BERT_RCNN 模型由知名的计算机科学家 Jacob Devlin 在 2018 年提出的。BERT_RCNN 模型是一种基于双向预训练语言模型（BERT）和循环神经网络（RNN）的两阶段的全加和深度神经网络模型，在第一阶段，通过使用 BERT 模型把输入文本编码为低维向量表征，这样从一定程度上把不同文本构成的输入矩阵转换为代表整个文档的局部特征。在第二阶段，RCNN 模型使用 BERT 局部特征和文档边界提取技术提取文档局部特征并进行分类。同时，BERT_RCNN 模型也通过在 BERT 上添加自定义层优化性能。

BERT_RCNN 模型结构由三个主要部分组成：BERT，循环神经网络和分类器。BERT 的作用是将文本转换为向量表示，以便后续的模型可以处理。循环神经网络（RNN）的作用是捕捉文本中的低维特征，以便模型可以更好地理解文本。分类器的作用是将文本分类为不同的类别。而且 BERT RCNN 模型可以有效地处理文本中的多义性。

七、 ENRIE 模型

ENRIE（Enhanced Language Representation with Informative Entities）模型由 Sun 等人在 2019 年提出，基于双向循环神经网络（Bi-RNN）[27]。ENRIE 模型的结构主要由四个部分组成：输入层、双向循环层、全连接层和输出层。输入层将文本转换为双向循环神经网络可以处理的形式，通常是一个矩阵，其中每一行代表一个单词，每一列代表一个特征，比如词性、词频等。双向循环层使用双向循环神经网络对输入层的输入进行循环操作。全连接层将双向循环层的输出进行整合，从而提取更多的文本特征。输出层使用 $softmax$ 函数将全连接层的输出转换为文本分类的结果。

ENRIE 基于自然语言的模型可以表示为节点和边图。每个节点代表一个概念或实体，每条边代表两个节点之间的关系。该模型在大量自然语言及其相应解释的语料库上进行训练，使用循环神经网络来识别节点之间的关系并为自然语言生成解释。ENRIE 模型的优势在于可以处理变长的文本和不同长度的文本，但是它也只能处理短文本，还存在梯度消失的问题，这导致模型的训练效果可能不佳。

ENRIE 模型由文本编码器 T-Encoder 和知识编码器 K-Encoder 两个模块构成。对于 K-Encoder，首先令两序列分别各自通过 multi-head self-attention 层，得到：

$$\begin{aligned}\left\{\tilde{\boldsymbol{w}}_1^{(i)},\ldots,\tilde{\boldsymbol{w}}_n^{(i)}\right\} &= \mathrm{MH-ATT}\left(\left\{\boldsymbol{w}_1^{(i-1)},\ldots,\boldsymbol{w}_n^{(i-1)}\right\}\right)\\ \left\{\tilde{\boldsymbol{e}}_1^{(i)},\ldots,\tilde{\boldsymbol{e}}_m^{(i)}\right\} &= \mathrm{MH-ATT}\left(\left\{\boldsymbol{e}_1^{(i-1)},\ldots,\boldsymbol{e}_m^{(i-1)}\right\}\right)\end{aligned} \tag{3.23}$$

将序列同相应的实体对齐后输入信息融合层，对于无对应实体的序列：

$$\begin{aligned}\boldsymbol{h}_j &= \sigma\left(\tilde{\boldsymbol{W}}_t^{(i)}\tilde{\boldsymbol{w}}_j^{(i)}+\tilde{\boldsymbol{b}}^{(i)}\right)\\ \boldsymbol{w}_j^{(i)} &= \sigma\left(\boldsymbol{W}_t^{(i)}\boldsymbol{h}_j+\boldsymbol{b}_t^{(i)}\right)\end{aligned} \tag{3.24}$$

对于有对应实体的序列：

$$\begin{aligned} \boldsymbol{h}_j &= \sigma\left(\tilde{\boldsymbol{W}}_t^{(i)}\tilde{\boldsymbol{w}}_j^{(i)} + \tilde{\boldsymbol{W}}_e^{(i)}\tilde{\boldsymbol{e}}_k^{(i)} + \tilde{\boldsymbol{b}}^{(i)}\right) \\ \boldsymbol{w}_j^{(i)} &= \sigma\left(\boldsymbol{W}_t^{(i)}\boldsymbol{h}_j + \boldsymbol{b}_t^{(i)}\right) \\ \boldsymbol{e}_k^{(i)} &= \sigma\left(\boldsymbol{W}_e^{(i)}\boldsymbol{h}_j + \boldsymbol{b}_e^{(i)}\right) \end{aligned} \tag{3.25}$$

最后引入 DEA 模型随机隐藏部分实体，要求模型从既定的实体序列中预测概率最大的实体，预测的概率分布计算式如下：

$$p\left(e_j \mid w_i\right) = \frac{\exp\left(\text{linear}\left(\boldsymbol{w}_i^o\right) \cdot \boldsymbol{e}_j\right)}{\sum_{k=1}^{m} \exp\left(\text{linear}\left(\boldsymbol{w}_i^o\right) \cdot \boldsymbol{e}_k\right)} \tag{3.26}$$

其中 linear 函数表示线性层，DEA 预测的损失函数可通过交叉熵计算。

第四章　网络舆情对股指期货收益率影响实证分析

作为网络上最具代表性的金融社交媒体，股吧成为投资者分享投资信息和情感的重要平台，也成为了研究投资者情绪的重要数据源。近年来，越来越多的研究将股吧的情绪分析应用到投资者情绪和市场预测中。熊艳（2022）分析了股吧中的情绪表达和投资者的行为，并发现情绪对投资者的行为产生了显著影响，股票价格的变动与情绪指数之间存在显著的相关性。谢雨桐（2020）的研究表明，股吧的情绪对投资者的情绪和决策有着显著的影响，其中负面情绪会导致投资者的决策更加保守。此外，股吧情绪还被应用到市场预测中，有研究使用了股吧中的情绪信息和股票历史数据进行预测，并获得了良好的效果。另外，范小云等（2022）使用机器学习方法对股吧的情绪进行分类，并将情绪分类结果应用到投资者情绪和市场预测中，获得了较好的结果。因此可以得出，股吧情绪对投资者的情绪和决策具有显著影响，并且可以被应用到市场预测中。

投资者情绪对金融市场存在较大影响，股指期货也不例外。白露莹（2021）研究发现，投资者情绪的波动与股指期货价格的波动密切相关，投资者情绪的上升可以导致股指期货价格的上升，而投资者情绪的下降则会导致股指期货价格的下降。其次，一些文献将焦点放在了投资者情绪对交易量的影响上。何芳（2021）研究发现，当投资者情绪高涨时，股指期货的交易量也会随之增加。这可能是因为情绪高涨的投资者更加积极地参与到股指期货交易中，并且愿意承担更高的风险。最后，还有一些文献研究了投资者情绪对市场流动性的影响。尹海员等（2019）研究发现，情绪波动大的投资者会导致股指期货市场流动性下降，因为他们往往会更加谨慎，更少地参与市场交易。综上所述，投资者情绪对股指期货价格、交易量和市场流动性均有较大的影响。

因此本文选取股吧文本对股指期货收益率影响进行研究。

第一节　网络舆情数据的获取与实验环境

一、 网络舆情数据的获取

本文选取的网络舆情数据来源于东方财富股指期货股吧的主题帖数据，获取方式为 Python 爬虫，主要通过 Requests 获取请求与响应，同时为了避开反爬虫机制，使用 UserAgent.random 不断更换响应头，最后将获取主题帖的发帖时间、阅读数、评论数和标题文本储存在本地，部分数据如表4.1所示，选取的发帖时间范围为 2018 年 1 月 3 日至 2023 年 1 月 20 日，共计 117245 条数据，词云图如图4.1所示。其中交易日的发帖数为 111349 条，非交易日发帖数为 5896 条，交易日发帖数占发帖总数的 94.97%，可见发帖数量与市场是否可交易相关。

将数据进行清洗后手动标注 5863 条数据作为基于金融文本的预训练任务的输入，其中情感标注分成 0，1，2 三类，0 表示消极情绪，看跌看空，1 表示没有明显的情感偏好或者是对未来走势不确定，2 表示积极情绪，看涨看多。

表 **4.1**　东方财富股指期货吧金融文本训练库（部分）

发帖时间	阅读数	评论数	标题文本	情感分类
2022-03-16	179	0	IC 当月连续 (SF060120) 在杀恐慌割肉盘	0
2018-10-22	3278	0	使劲拉，再涨一百个点，空方全爆	2
2020-07-16	300	0	神创板也撑不住了啊，一百多倍市盈率啊	1
2019-01-04	299	0	创业板注册制次新股见底机会多多	2
2020-02-05	543	0	跳水开始	0
2021-10-26	350	2	卖出一切股票，股指期货继续做空	0
2019-12-23	563	0	管理层不想看到的	1
2020-08-24	478	0	首先铁门栓的估计是太平洋	1

图 **4.1**　股指期货吧主题帖词云图

对于股指期货的数据，本文从天软（Tinysoft）数据库中获取了 IF00、IH00 和 IC00 三种股指期货主力合约的开盘价、收盘价、最高价、最低价、昨收盘价、成交量等指标，作为后文情绪指标和股指期货收益率相关性测试以及基于时间序列 LSTM 模型的股指期货价格预测两个部分的数据支撑。

表 **4.2**　实验环境

环境	配置
操作系统	Windows 11
CPU	Intel(R) Core(TM) i7-12700H
GPU	RTX 3070Ti Laptop
语言环境	Python 3.9
深度学习框架	Pytorch 1.13.1

二、 实验环境

本文进行深度学习的实验环境如下表所示（表4.2）：

第二节　基于股吧文本的 BERT 预训练过程

目前已经训练好的 BERT 预训练模型有很多，比较常见的预训练模型包括：BERT-Base（12 层双向的小 Transformer 网络）、BERT-Large（24 层双向的大 Transformer 网络）、RoBERTa（Facebook AI 团队基于 BERT-Base 训练的）、ALBERT（Google AI 团队改进的高效轻量级 BERT 模型）、DistilBERT（Hugging Face 团队改进的高效轻量级 BERT 模型）等。本文采用的是哈工大和科大讯飞共同开发训练的 chinese-bert-wwm-ext 模型，该模型在预训练过程中使用了大量的中文语料，并使用 BERT 模型的架构和 MLM 任务进行预训练，具有较高的语言理解能力与很好的语言基础知识，在中文 NLP 任务中表现出了较好的性能，是一个值得推广的中文预训练模型。通过对获取的东方财富股指期货吧文本对该模型进行预训练，具体实现过程如下：

对于输入的一段文本 $T = [w_1, w_2, \ldots, w_{n-1}, w_n]$，在经过 Bert 的 Encoder 层（通常采取默认 12 层的 Transformer Encoder Layer）的编码后进行 MLM（Masked Language Model）与 NSP（Next Sentence Prediction）任务。对于 MLM 任务，其目的是通过让模型预测被掩盖的单词，来学习语言表达。首先是数据预处理，即对于每一个输入的句子进行切分，通过模型自带的词汇表将其转换为向量，同时给句首添加 <CLS> 符号，两句句子中间添加 <SEP> 符号，句末也添加 <SEP> 符号，部分文本切分结果如表4.3所示。

并对列表中的一些单词进行标记，表示这些单词需要被掩盖。之后对已切分好的句子词汇表（Tokenizer）进行编码转化为向量，部分文本编码展示如表4.4所示。

表 **4.3**　部分文本切分展示

输入文本	输出切分后结果
商品期货的保证金提高了百分之五十	[CLS] 商品期货的保证金提高了百分之五十 [UNK] [SEP]
非常好，跳水的预试，下午暴跌。	[CLS] 非常好，跳水的预试，下午暴跌。[SEP]
期指主力合约全跌，IF2104 跌幅 0.78%	[CLS] 期指主力合约全跌，IF 2104 跌幅 0 . 78 [UNK] [SEP]
主力你侧重去做多金与银可引发美元下跌股市反弹！	[CLS] 主力你侧重去做多金与银可引发美元下跌股市反弹！[SEP]
主力既做空赚钱，又打压股价吃进底位筹码拿货，一箭双雕	[CLS] 主力既做空赚钱，又打压股价吃进底位筹码拿货，一箭双雕 [SEP]
国庆节后利好一般都比较多，我们一起多一多，乐呵呵	[CLS] 国庆节后利好一般都比较多，我们一起多一多，乐呵呵 [SEP]

表 **4.4**　部分文本编码展示

输入文本	输出映射后结果
商品期货的保证金提高了百分之五十	[101 1555 1501 3309 6573 4638 924 6395 7032 2990 7770 749 4636 1146 722 758 1282 100 102]
非常好，跳水的预试，下午暴跌。	[101 7478 2382 1962 8024 6663 3717 4638 7564 6407 8024 678 1286 3274 6649 102]
期指主力合约全跌，IF2104 跌幅	[101 3309 2900 712 1213 1394 5276 1059 6649 8024 8898 8650 9099 6649

接下来是采样，遍历完所有已被切分编码后的文本，传入列表 input_ids。之后对每个单位随机选择 15% 的位置进行标记，对于被标记的部分 80% 替换为“<mask>”，10% 替换为词汇表中的随机词元，10% 保留不变，并且将被替换部分的位置处标记为 0，其余部分为 1，传入列表 input_masks。以“主力既做空赚钱，又打压股价吃进底位筹码拿货，一箭双雕”为例：

编码后的句子为：[101 712 1213 3188 976 4958 6611 7178 8024 1348 2802 1327 5500 817 1391 6822 2419 855 5040 4772 2897 6573 8024 671 5055 1352 102]

替换后的句子向量：[-100 -100 -100 -100 -100 -100 -100 -100 -100 -100 2802 1327 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100 -100]

句子标记向量为：[1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1]。

第三节　模型效果评价与结果呈现

通常对于分类模型效果的评价来说，精确率、召回率、F1 得分与预测准确率可以比较客观的呈现效果。精确率（Precision）：指分类器预测为正样本的实际正样本数与预测正样本数的比值，精确率高表示分类器预测为正样本的样本中，实际正样本的比例较大，预测效果较好。召回率（Recall）：指分类器预测为正样本的实际正样本数与实际正样本总数的比值，召回率高表示分类器识别正样本的能力较强，遗漏较少。F1：精确率和召回率的调和平均数，F1 值反映了精确率和召回率的平衡，通常用于评估分类器的性能，其值越接近 1 表示分类器的性能越好。准确率（Accuracy）：分类器在所有样本中预测正确的比例。这些指标通常被用于评估分类模型的性能，不同的指标可以反映出模型在不同方面的表现。例如，精确率和召回率可以告诉我们模型在识别正例和负例时的表现如何，而 F1 值则可以综合考虑这两个指标，准确率则可以告诉我们模型在整体上的表现如何。四者分别定义如下：

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{4.1}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{4.2}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4.3}$$

$$Accuracy = \frac{TruePositives + TrueNegatives}{TruePositives + TrueNegatives + FalsePositives + FalseNegatives} \tag{4.4}$$

对于上文提到的 11 个文本情绪识别模型，其中 BERT 模型分为已预训练好的以及基于股吧文本的预训练 BERT，测试集上计算出的 F1 得分与准确率结果如表4.5所示。可以看出，非预训练深度学习情绪识别模型的预测效果相较于基于预训练的深度学习情绪识别模型预测效果有着一定的差距，这是因为非预训练的深度学习模型没有使用大规

表 **4.5**　模型表现结果

模型	Accuracy	F1
TextCNN	0.7881	0.7794
TextRNN	0.7873	0.7762
TextRCNN	0.7931	0.7891
TextRNN_*Att*	0.7875	0.7778
Transformer	0.7662	0.7516
BERT(Pre_trained)	0.8105	0.8041
BERT	**0.8126**	**0.8070**
BERT_CNN	0.8062	0.7938
BERT_DPCNN	0.8086	0.7953
BERT_RNN	0.8092	0.7965
BERT_RCNN	0.8101	0.8036
ENRIE	0.8116	0.8054

模的未标注语料进行预训练，因此在许多情况下需要更多的标注数据来进行训练，并且容易受到数据集的噪声影响。其中 TextCNN、TextRNN、TextRCNN 和 TextRNN_*Att* 表现相近，使用了更高维度的循环神经网络的 TextRCNN 表现最好，准确率为 79.31%，F1 值为 0.7891。

而基于预训练的深度学习情绪识别模型可以利用大量的未标注数据进行预训练，从而获得更好的文本表示能力。通过在有标注数据上进行微调，可以将预训练模型迁移到情感分析等任务中，可以获得更好的泛化性能，即使在数据集较小的情况下也可以取得较好的表现。其中表现最好的是基于股吧文本的预训练 BERT 模型，准确率达到了 81.26%，F1 值为 0.8070，其次是 ENRIE 模型和已预训练好的 BERT 模型。BERT 模型之所以能以较高的精确率进行预测，主要归功于其大量预训练后获得了相应的先验知识，在情绪识别提取分类任务上具有更好的可迁移性和泛化性能，在此基础上使用金融文本对其进行进一步的预训练，可以实现更好的情感分类效果。基于股吧文本的预训练 BERT 模型准确率、F1 值和模型交叉熵损失变化拟合图如图4.2和4.3所示。

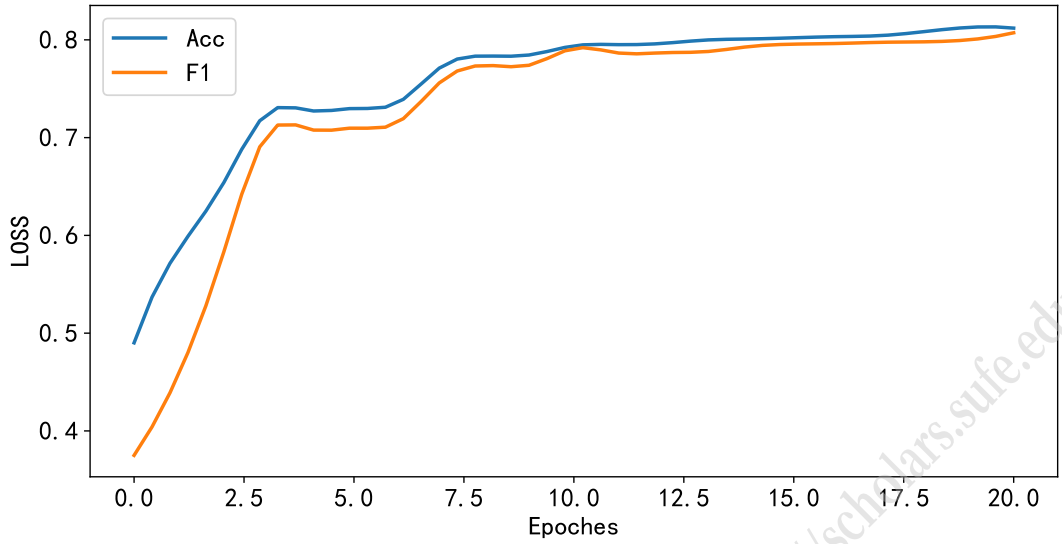

图 **4.2**　基于股吧文本的预训练 **BERT** 模型准确率和 **F1** 值变化拟合图

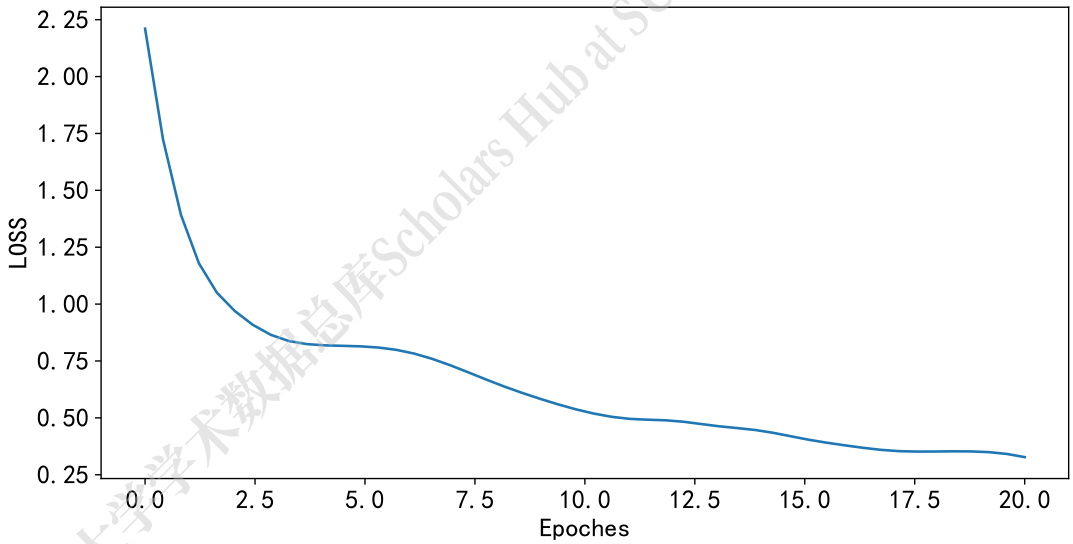

图 **4.3**　基于股吧文本的预训练 **BERT** 模型交叉熵损失变化拟合图

第四节　网络舆情情绪指标构建

基于上一节的结果，本文采取基于股吧文本的预训练 BERT 模型对除训练集外剩下未标记的数据进行情感分类，得到标记为 0，即消极情绪的帖子数量为 45139 条，标记为 1 的中立帖子数为 40653，标记为 2 的积极情绪主题帖有 31392 个。将所有数据情感标注完后将所有非交易日数据前移至上一交易日，并依此构建情绪指标。使用 Antweiler[7]

对雅虎上股票留言板信息构建情绪指标的方法

$$B_t \equiv \ln(\frac{1+M_t^{BUY}}{1+M_t^{SELL}}) \tag{4.5}$$

其中 $M_t^c \equiv \sum_{i\in\mathcal{D}(t)} w_i x_i^c$，指标变量 $c \in \{\text{BUY}, \text{HOLD}, \text{SELL}\}$，$M_t^{BUY}$ 表示正向情绪或看涨看多的帖子数量，而 M_t^{SELL} 则表示负面情绪或看跌看空的帖子数量。B_t 代表第 t 天对于情绪的测度，作为情绪指标。之后使用 $Z-score$ 标准化对计算出的情绪指标进行处理，即

$$Z = \frac{X-\mu}{\sigma} \tag{4.6}$$

其中 μ 为样本的平均值，σ 为样本的标准差，这样可以清除掉数据中的极端值。

构建好情绪指标后，绘制 IF00 收益率与情绪指标折线图，如图4.4所示。

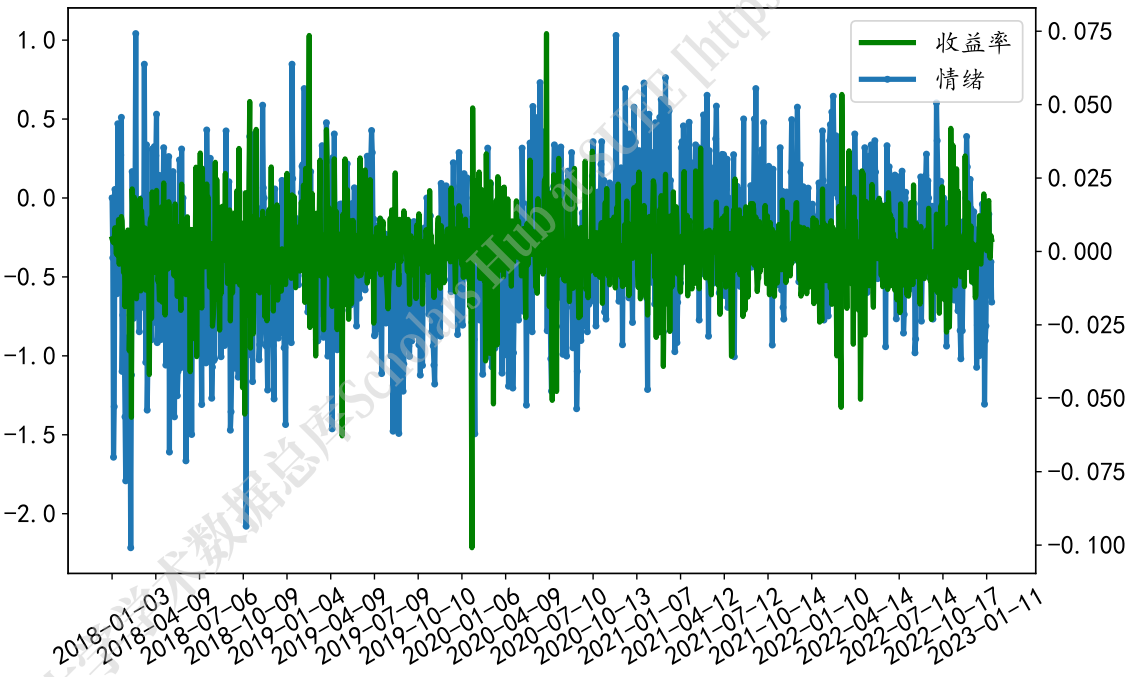

图 **4.4**　**IF00** 收益率与情绪指标折线图

为了验证相关性，绘制 IF00、IH00 和 IC00 三个股指期货主力合约与情绪指标滞后相关系数图，如图4.5、4.6和4.7所示。

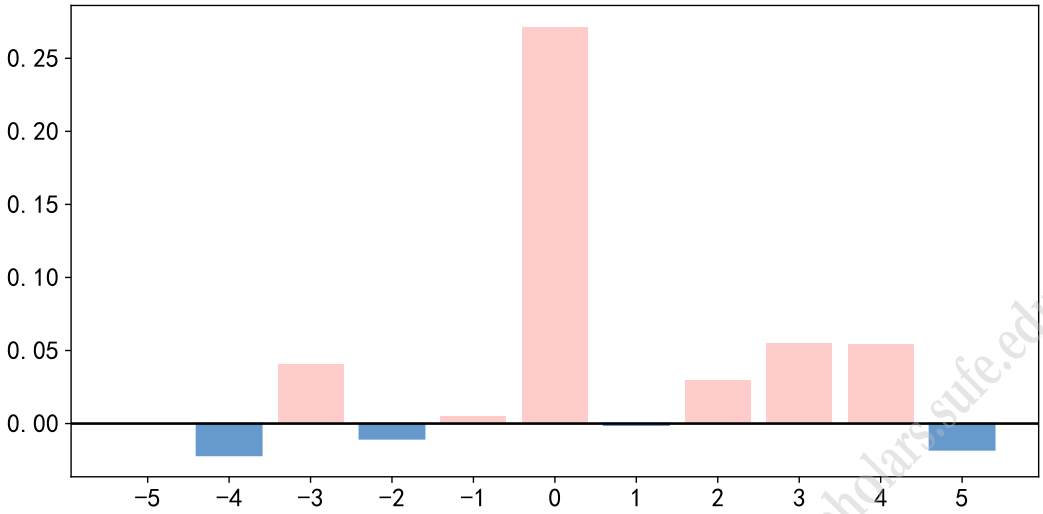

图 **4.5**　**IF00** 收益率与情绪指标滞后相关系数图

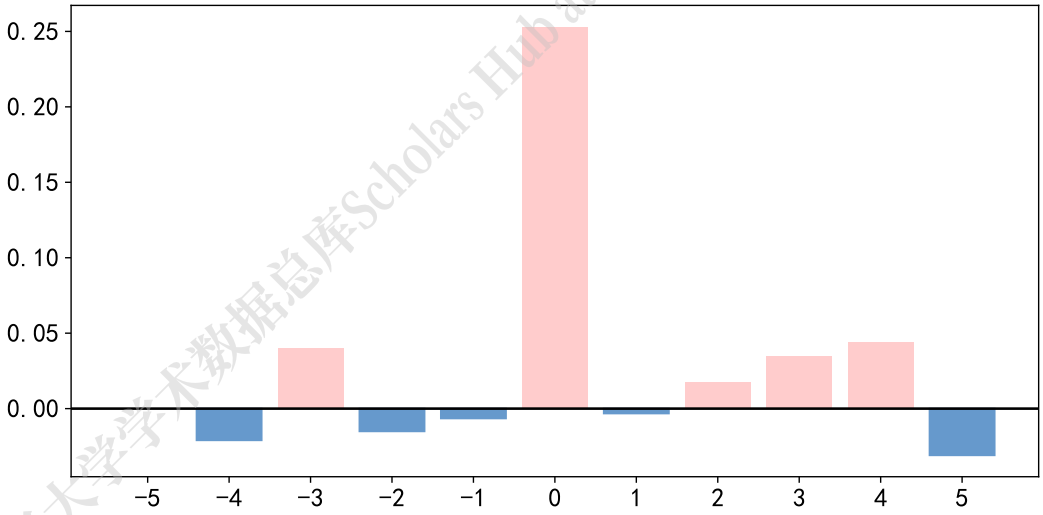

图 **4.6**　**IH00** 收益率与情绪指标滞后相关系数图

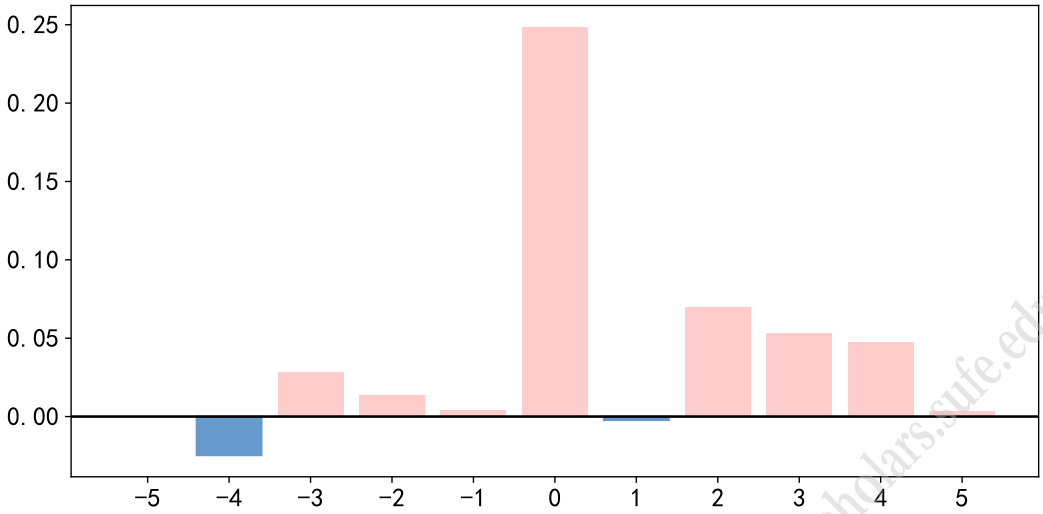

图 **4.7**　**IC00** 收益率与情绪指标滞后相关系数图

可以看到，情绪指标与 IF00、IH00 和 IC00 的相关系数分别达到了 0.2715、0.2530 和 0.2486。

第五节　基于时间序列 LSTM 模型的股指期货价格预测

通过以上分析可以发现，情绪指标与股指期货主力合约收益率存在着相关性，因此本文将情绪指标纳入对于股指期货价格预测模型中，采用的是基于时间序列的 LSTM 模型进行预测。选择的自变量主要有量价指标，包括前 20 个交易日的开盘价、收盘价、最高价、最低价、昨收盘价、成交量等，除此之外还加入情绪指标，具体见表4.6：

表 **4.6**　**LSTM** 预测模型输入自变量

自变量	符号
开盘价	$Open$
收盘价	$Close$
最高价	$High$
最低价	Low
昨收盘价	Y_Close
成交量	Vol
情绪指标	$Sentiment$

为了进一步对比情绪指标对股指期货主力合约收益率的预测能力，本文分别训练了两个基于时间序列 LSTM 模型，其中一个加入情绪指标，另一个不加入。将所有指标进行最大最小归一化，即

$$X' = \frac{X - X_{\max}}{X_{\max} - X_{\min}} \tag{4.7}$$

之后选取总回测区间的 80% 作为训练集，20% 作为测试集，对 IF00 即沪深 300 股指期货主力合约收盘价进行预测，模型表现结果如下（见表4.7、图4.8和图4.9）

表 **4.7**　模型表现结果

模型	RMSE
无情绪指标的 LSTM	0.0829
有情绪指标的 LSTM	0.0681

可以看到，加入了情绪指标后，LSTM 模型的均方误差从 0.0829 下降至 0.0681，下降了 17.85%，表现有着较为显著的提升，从图中也可以看出预测值与真实值的拟合效

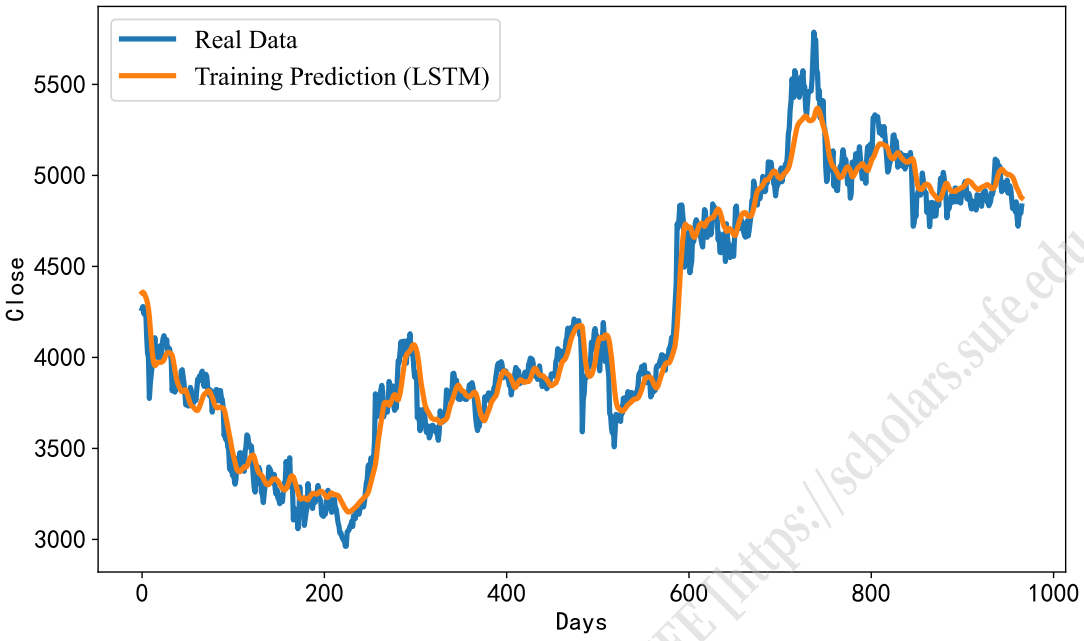

图 **4.8**　无情绪指标的 **LSTM** 预测结果

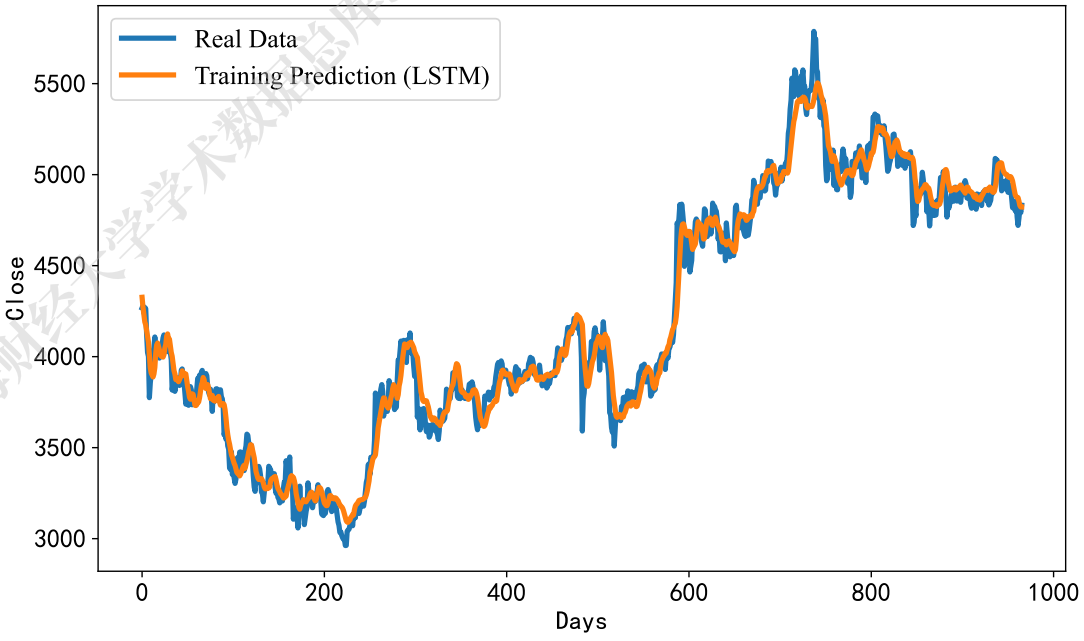

图 **4.9**　有情绪指标的 **LSTM** 预测结果

果更好。最后绘制出 IF00 全区间 LSTM 预测结果，如图4.10所示。

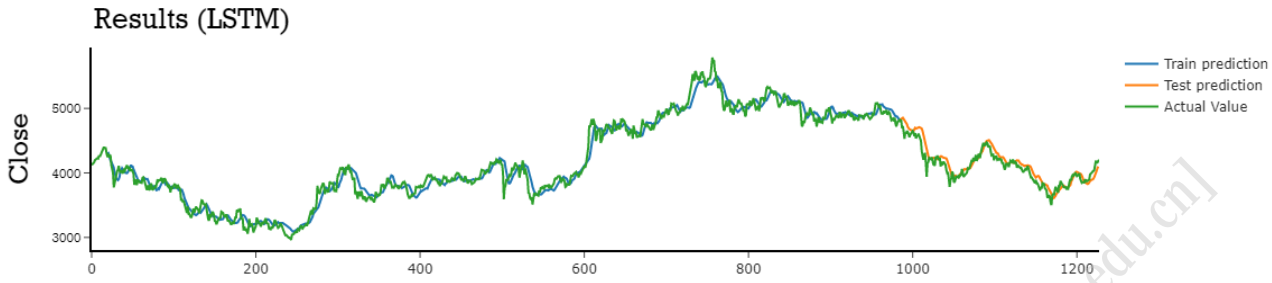

图 **4.10**　全区间 **LSTM** 预测结果

第五章　结论与展望

第一节　研究结论

随着互联网和信息技术的不断发展，网络舆情在金融领域的影响力逐渐扩大。从早期阶段的金融和网络舆情是两个独立的领域，金融行业主要依靠传统的市场研究和分析方法，网络舆情则主要用于社会事件的传播和评论；到信息化时代的金融机构开始建立自己的网站和社交媒体账号，通过发布各种信息来引导市场预期的同时，网络舆情分析技术也逐渐成熟，可以对社交媒体等平台上的言论进行实时监测和分析；之后是大数据时代。近年来，随着大数据技术的发展，金融和网络舆情之间的关联越来越密切。金融机构可以通过分析社交媒体上的大数据，掌握市场情绪和投资者心理，提高投资决策的准确性。同时，大数据技术也为网络舆情分析提供了更加精准的工具和方法。智能化则是未来发展的方向，近年来，人工智能和自然语言处理技术的不断发展，为金融与网络舆情的融合提供了更加广阔的空间，例如，基于深度学习的情感分析技术可以更加准确地识别和分析网络舆情的情感倾向，从而更加精准地预测市场走势和波动趋势，同时，智能化技术也为金融监管提供了更加科学和精准的手段。

本文通过爬虫获取东方财富股指期货股吧 5 年的主题帖数据以及天软数据库获取的股指期货日频数据，基于预训练的深度学习模型实现情感识别分类任务进行了如下的研究：

（1）从获取的东方财富股指期货股吧的主题帖的时间分布来看，在 117245 条数据中，交易日数据有 111349 条，交易日发帖数占发帖总数的 94.97%，股民在交易日的发帖意愿远高于非交易日。从文本来看，悲观唱空的消极情绪占比较大，这也可以从之后标注完的全数据集情感可以看出（45139 条消极情绪主题帖，40653 条中立主题帖和 31392 条积极情绪主题帖）

（2）将获取的股吧文本对 BERT 模型的 MLM 子任务上进行一次预训练，使得已经训练好的 BERT 模型在挖掘与识别金融文本的任务中有更好的表现。之后对比了非预训练的深度学习模型和基于预训练的深度学习模型的情绪识别能力，一共构建了 11 个模型，发现适当增加了复杂程度的 TextRCNN 是非预训练深度学习模型中表现最好的，准确率达到 79.31%，F1 值为 0.7891。表现最好的是基于股吧文本的预训练 BERT 模型，准确率达到了 81.26%，F1 值为 0.8070，其次是 ENRIE 模型和已预训练好的 BERT 模型，结果说明了对金融文本的预训练能为 BERT 模型带来一定的先验知识，并且该结论可推广至任何领域，即先要对该领域的文本分类进行一次预训练。最后，相较于 Transformer 模型，BERT 在文本情绪识别分类任务的表现是更优的。

（3）基于股吧文本的预训练 BERT 模型构建的情绪指标与三种股指期货主力合约收益率的相关系数分别达到了 0.2715、0.2530 和 0.2486，进一步说明了网络舆情与股指期货的收益率之间存在的相关性。使用基于时间序列 LSTM 模型预测股指期货价格时，情

绪指标的加入能提升模型预测的准确率，均方误差从未加入情绪指标的 0.0829 降低至 0.0681，减少了 17.85% 有着较为显著的提升，可以将情绪指标纳入深度学习预测任务中，以更高的准确率对股指期货价格进行预测。

基于以上结论，网络舆情对金融市场有着比较重要的影响，其中的股指期货市场也不例外，由此带来的启发是在信息化与智能化的时代，把握好网络舆情十分重要。对于投资者而言，关注网络舆情对金融市场的影响非常重要，要做到及时监测网络舆情，了解市场情绪。投资者可以通过搜索引擎、社交媒体、新闻网站等途径获取相关信息，利用情感分析工具对情绪进行分析和量化。但是需要注意，网络舆情只是影响市场的因素之一，投资者需要结合其他指标（如经济数据、公司财报等）进行综合分析，保持理性和冷静。网络舆情往往存在误导性和偏颇性，投资者应该保持理性和冷静，不要被过度渲染或误导，有时候可以把握机会，考虑逆势而为：当网络舆情对某个股票或行业的情绪为负面时，投资者可以考虑逆势而为，把握低买高卖的机会。

对于政策制定者来说，关注网络舆情可以帮助他们更好地制定和调整金融政策：（1）建立网络舆情监测系统：政策制定者可以建立一个系统来监测网络舆情的变化，并分析它们对金融市场的影响，可以帮助他们更早地发现市场的风险和机会，并及时采取相应的措施。（2）加强舆情分析和研究：政策制定者可以加强对网络舆情的分析和研究，探究它们对金融市场的影响机制和规律，进而制定更有针对性的政策，同时，政策制定者也可以加强与学术界和业界的合作，利用大数据和人工智能等技术手段进行深入研究。（3）制定应对措施：一旦发现网络舆情对金融市场产生了不良影响，政策制定者应该采取相应的措施加以应对，包括及时发布官方信息、加强监管力度、引导市场预期等，从而减少不必要的市场波动。（4）宣传政策和理念：政策制定者还可以通过加强宣传和引导网络舆情，向公众传递正确的金融理念和政策，促进金融市场的稳定和健康发展。

第二节　不足与展望

由于时间、设备与技术多方面的限制，本文所研究的过程也存在着一定的不足之处：

（1）获取数据的频率较高，且数据细分指标并未充分利用。本次研究获取的股吧文本数据其中有精确到秒的发帖时间，获取的股指期货数据也是日频，受限于技术和设备问题并未在比交易日更高频率的框架下进行网络舆情与股指期货收益率的高频研究。此外，获取股吧数据时，每个主题帖有对应的阅读数与评论数，这两个指标越高代表发帖者越大概率是意见领袖，其对于当日情绪的影响权重必然更大，本次研究并未用到。

（2）本次研究使用的 BERT 已预训练好的模型是哈工大与科大讯飞联合训练的 BERT-wwm-ext 模型，随着近两年 BERT 模型的研究，越来越多的机构推出了训练规模更大，过程更为复杂，参数更多，成本更高的 BERT 模型，受限于技术与设备，本次研究并未用最新的研究成果，如果使用了这些可能可以进一步提升文本分类准确率等结果。

（3）目前有很多不同的情绪指标构建方法，本次研究使用了由 Antweiler 提出较为经典的指标，实际上有更多指标可以进行尝试，通过对比不同指标的预测效果可以进行更深层面的研究，同样地在预测股指期货价格时可以使用不同模型对比预测效果，这也是可以进一步深入研究的方向。

参考文献

[1] Aboody D, EvenTov O, Lehavy R, et al.Overnight returns and firmspecific investor sentiment[J].Journal of Financial and Quantitative Analysis.2018,53(2):485−505.

[2] Antweiler W, Frank M Z. Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards[J]. Journal of Finance, 2004, 59(3).

[3] Baker M, Wurgler J. Investor Sentiment and the Cross-Section of Stock Returns [J]. Wiley, 2006(4).

[4] Cedric Mbanga,Ali F. Darrat,Jung Chul Park. Investor sentiment and aggregate stock returns: the role of investor attention[J]. Review of Quantitative Finance and Accounting,2019,53(2).

[5] Chunpeng Yang,Wei Yan. Does High Sentiment Cause Negative Excess Return?[J]. International Journal of Digital Content Technology and its Applications,2011,5(12).

[6] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J]. 2018.

[7] Gregory W. Brown, Michael T. Cliff.Investor sentiment and the near-term stock market[J].Journal of Empirical Finance.2004,11(1).

[8] Hu Z, Liu W, Bian J, et al. Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction[J]. 2017.

[9] Johan Bollen,Huina Mao,Xiaojun Zeng. Twitter mood predicts the stock market[J]. Journal of Computational Science,2011,2(1).

[10] Johnson R, Tong Z. Deep Pyramid Convolutional Neural Networks for Text Categorization[C].2017.

[11] Kim Y. Convolutional Neural Networks for Sentence Classification[J]. Eprint Arxiv, 2014.

[12] Lai S, Xu L, Liu K, et al. Recurrent Convolutional Neural Networks for Text Classification[C]. 2015.

[13] Lee C, Shleifer A, Thaler R. Investor sentiment and the closed-end fund puzzle[J]. Journal of Finance, 1991, 46(1):75-109.

[14] Lin, Chu-Bin, Chou, et al. Investor sentiment and price discovery: Evidence from the pricing dynamics between the futures and spot markets[J]. Journal of Banking &, 2018, 90(May):17-31.

[15] Lin Peng, Wei Xiong. Investor attention, overconfidence and category learning[J]. Journal of Financial Economics,2005,80(3).

[16] Liu P, Qiu X, Huang X. Recurrent Neural Network for Text Classification with Multi-Task Learning. 2016.

[17] Malcolm Baker, Jeremy C Stein. Market liquidity as a sentiment indicator[J]. Journal of Financial Markets,2003,7(3).

[18] Peng Z, Wei S, Tian J, et al. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification[C]. 2016.

[19] Sun Y, Wang S, Li Y, et al. ERNIE: Enhanced Representation through Knowledge Integration[J]. 2019.

[20] Vaswani A, Shazeer N, Parmar N, et al. Attention Is All You Need[J]. arXiv, 2017.

[21] Zimbra D, Fu T, Li X. Assessing Public Opinions Through Web 2.0: A Case Study on Wal-Mart[C].International Conference on Information Systems, Icis 2009, Phoenix, Arizona, Usa, December, 2009: 67.

[22] 白露莹. 投资者情绪对股指期货定价偏差的影响研究 [D]. 四川大学,2021.000439.

[23] 何芳. 投资者情绪对股票市场表现的影响研究 [D]. 湖南大学,2021.000660.

[24] 范小云, 王业东, 王道平, 郭文璇, 胡煊翊. 不同来源金融文本信息含量的异质性分析——基于混合式文本情绪测度方法 [J]. 管理世界,2022.0145.

[25] 赖星武. 网络舆情对黄金期货价格的影响及预测研究 [D]. 上海财经大学,2021.

[26] 李思龙, 金德环, 李岩. 网络社交媒体提升了股票市场流动性吗?——基于投资者互动视角的研究 [J]. 金融论坛,2018,23(07):35-49+63.

[27] 龙文, 毛元丰, 管利静, 崔凌逍。财经新闻的话题会影响股票收益率吗？——基于行业板块的研究 [J]. 管理评论,2019,31(05):18-27.

[28] 孙明宏. 基于 BERT 模型的金融新闻舆情对股票收益率影响研究 [D]. 上海财经大学,2021.

[29] 谢雨桐. 投资者情绪与股票短期收益率的再讨论 [D]. 西南财经大学,2020.001567.

[30] 熊艳. 论坛发帖与股价行为: 情绪宣泄还是信息传递?[J]. 中央财经大学学报,2022(05):29-45

[31] 尹海员, 吴兴颖. 投资者日度情绪、超额收益率与市场流动性——基于 DCC-GARCH 模型的时变相关性研究 [J]. 北京理工大学学报 (社会科学版),2019,21(05):76-87+114.

[32] 张强, 杨淑娥, 杨红. 中国股市投资者情绪与股票收益的实证研究 [J]. 系统工程,2007(07):13-17.

[33] 朱南丽, 邹平, 张永平, 李学术, 杨琳琳, 张杨. 基于博客/微博信息量的投资者关注度测量研究——来自中国股票市场的经验数据 [J]. 经济问题探索,2015(02):159-166.

附录 A 构建并训练 BERT 模型

```
class Bert_Model(nn.Module):
def __init__(self, bert_path, classes=10):
  super(Bert_Model, self).__init__()
  self.config = BertConfig.from_pretrained(bert_path) # 导入模型超参数
  self.bert = BertModel.from_pretrained(bert_path) # 加载预训练模型权重
  self.fc = nn.Linear(self.config.hidden_size, classes) # 直接分类

def forward(self, input_ids, attention_mask=None, token_type_ids=None):
  outputs = self.bert(input_ids, attention_mask, token_type_ids)
  out_pool = outputs[1] # 池化后的输出 [bs, config.hidden_size]
  logit = self.fc(out_pool) # [bs, classes]
  return logit

def get_parameter_number(model):
  #  打印模型参数量
  total_num = sum(p.numel() for p in model.parameters())
  trainable_num = sum(p.numel() for p in model.parameters() if p.requires_grad)
  return 'Total parameters: {}, Trainable parameters: {}'.format(total_num, trainable_num)

DEVICE = torch.device("cuda" if torch.cuda.is_available() else "cpu")
EPOCHS = 5
model = Bert_Model(bert_path).to(DEVICE)
print(get_parameter_number(model))
optimizer = AdamW(model.parameters(), lr=2e-5, weight_decay=1e-4) #AdamW优化器
scheduler = get_cosine_schedule_with_warmup(optimizer, num_warmup_steps=len(train_loader),
                           num_training_steps=EPOCHS*len(train_loader))

# 评估模型性能，在验证集上
def evaluate(model, data_loader, device):
model.eval()
val_true, val_pred = [], []
with torch.no_grad():
  for idx, (ids, att, tpe, y) in (enumerate(data_loader)):
    y_pred = model(ids.to(device), att.to(device), tpe.to(device))
    y_pred = torch.argmax(y_pred, dim=1).detach().cpu().numpy().tolist()
    val_pred.extend(y_pred)
    val_true.extend(y.squeeze().cpu().numpy().tolist())

return accuracy_score(val_true, val_pred) # 返回accuracy


# 测试集没有标签，需要预测提交
def predict(model, data_loader, device):
model.eval()
```

```
    val_pred = []
    with torch.no_grad():
      for idx, (ids, att, tpe) in tqdm(enumerate(data_loader)):
       y_pred = model(ids.to(device), att.to(device), tpe.to(device))
       y_pred = torch.argmax(y_pred, dim=1).detach().cpu().numpy().tolist()
       val_pred.extend(y_pred)
    return val_pred


def train_and_eval(model, train_loader, valid_loader,
                optimizer, scheduler, device, epoch):
    best_acc = 0.0
    patience = 0
    criterion = nn.CrossEntropyLoss()
    for i in range(epoch):
      """训练模型"""
      start = time.time()
      model.train()
      print("***** Running training epoch {} *****".format(i + 1))
      train_loss_sum = 0.0
      for idx, (ids, att, tpe, y) in enumerate(train_loader):
       ids, att, tpe, y = ids.to(device), att.to(device), tpe.to(device), y.to(device)
       y_pred = model(ids, att, tpe)
       loss = criterion(y_pred, y)
       optimizer.zero_grad()
       loss.backward()
       optimizer.step()
       scheduler.step() # 学习率变化

       train_loss_sum += loss.item()
       if (idx + 1) % (len(train_loader) // 5) == 0: # 只打印五次结果
         print("Epoch {:04d} | Step {:04d}/{:04d} | Loss {:.4f} | Time {:.4f}".format(
          i + 1, idx + 1, len(train_loader), train_loss_sum / (idx + 1), time.time() - start))
         # print("Learning rate = {}".format(optimizer.state_dict()['param_groups'][0]['lr']))

      """验证模型"""
      model.eval()
      acc = evaluate(model, valid_loader, device) # 验证模型的性能
      ## 保存最优模型
      if acc > best_acc:
       best_acc = acc
       torch.save(model.state_dict(), "best_bert_model.pth")

      print("current acc is {:.4f}, best acc is {:.4f}".format(acc, best_acc))
      print("time costed = {}s \n".format(round(time.time() - start, 5)))
```

附录 B　LSTM 模型预测股指期货价格

```
def split_data(stock, lookback):
    data_raw = stock.to_numpy()
    data = []

    # you can free play（seq_length）
    for index in range(len(data_raw) - lookback):
        data.append(data_raw[index: index + lookback])

    data = np.array(data);
    test_set_size = int(np.round(0.2 * data.shape[0]))
    train_set_size = data.shape[0] - (test_set_size)
    x_train = data[:train_set_size,:-1,:]
    y_train = data[:train_set_size,-1,0:1]

    x_test = data[train_set_size:,:-1,:]
    y_test = data[train_set_size:,-1,0:1]

    return [x_train, y_train, x_test, y_test]

lookback = 20
x_train, y_train, x_test, y_test = split_data(price, lookback)
import torch
import torch.nn as nn

x_train = torch.from_numpy(x_train).type(torch.Tensor)
x_test = torch.from_numpy(x_test).type(torch.Tensor)
y_train_lstm = torch.from_numpy(y_train).type(torch.Tensor)
y_test_lstm = torch.from_numpy(y_test).type(torch.Tensor)
y_train_gru = torch.from_numpy(y_train).type(torch.Tensor)
y_test_gru = torch.from_numpy(y_test).type(torch.Tensor)

class LSTM(nn.Module):
    def __init__(self, input_dim, hidden_dim, num_layers, output_dim):
        super(LSTM, self).__init__()
        self.hidden_dim = hidden_dim
        self.num_layers = num_layers

        self.lstm = nn.LSTM(input_dim, hidden_dim, num_layers, batch_first=True)
        self.fc = nn.Linear(hidden_dim, output_dim)

    def forward(self, x):
        h0 = torch.zeros(self.num_layers, x.size(0), self.hidden_dim).requires_grad_()
        c0 = torch.zeros(self.num_layers, x.size(0), self.hidden_dim).requires_grad_()
        out, (hn, cn) = self.lstm(x, (h0.detach(), c0.detach()))
```

```
        out = self.fc(out[:, -1, :])
        return out

model = LSTM(input_dim=input_dim, hidden_dim=hidden_dim, output_dim=output_dim,
    num_layers=num_layers)
criterion = torch.nn.MSELoss()
optimiser = torch.optim.Adam(model.parameters(), lr=0.001)

import time

hist = np.zeros(num_epochs)
start_time = time.time()
lstm = []

for t in range(num_epochs):
    y_train_pred = model(x_train)

    loss = criterion(y_train_pred, y_train_lstm)
    print("Epoch ", t, "MSE: ", loss.item())
    hist[t] = loss.item()

    optimiser.zero_grad()
    loss.backward()
    optimiser.step()

training_time = time.time()-start_time
print("Training time: {}".format(training_time))
predict = pd.DataFrame(scaler.inverse_transform(y_train_pred.detach().numpy()))
original = pd.DataFrame(scaler.inverse_transform(y_train_lstm.detach().numpy()))
```

致谢

本文至此基本结束，两年的研究生生活是自己人生中非常重要的组成部分，在此期间自己的能力与见识都有了很大的提升，随着学生时代的结束，内心也是百感交集。

首先要感谢我的导师，他也是我们班级的导师，在日常的学习和生活方面给予了我很大的帮助与支持。在导师的实践项目组中，我从头到尾参与了从无到有的量化投资项目，在充分地与业界结合的过程中，自己的编程能力也得到了很大的提升。导师提供了服务器，数据库等设备供我日常学习研究使用，在论文选题、内容与修改参考意见方面都为我提供了详细的建议，耐心指导，缺少研究所需数据时也联系业界为我提供相应的数据库，让我可以完成学位论文的撰写。在此，衷心地向我的导师致以感谢！也感谢所有在我研究生期间教过我的老师！

在日常的项目组交流过程中，实践项目组的同学们互帮互助，共同协作完成项目，在日常组会上对我论文选题与结构都提出了中肯的意见，对我有着相当重要的参考价值，在此同样非常感谢各位同学对我学习生活与学位论文完成上提供的帮助。此外也非常感谢我的室友，在我遇到不懂的编程问题时候十分耐心的教导，为我的职业规划与日常工作完成方面给予了很多建议，是我的良师益友。

同时我也要对在评审过程中对我论文提出修改意见的三位老师表达感谢，他们从自身教学与从业经历出发，为我论文提出了相应的修改意见，让本文更加充实完善，结构更加清晰。

最后我要感谢父母对我生活、学习和精神方面的支持，也要感谢共同学习成长的同学们！

个人简历及在学期间发表的研究成果

个人简历：

徐英皓，江西南昌人，1999 年 11 月 10 日生。

2017 年 9 月至 2021 年 6 月，本科就读于南昌大学经济统计学专业。

2021 年 9 月至 2023 年 6 月，硕士就读于上海财经大学金融科技专业。

Shanghai University of Finance and Economics

硕士学位论文

上海财经大学

上 海 财 经 大 学

地　　址：中国上海国定路 777 号　　777 Guo Ding Road

邮　　编：200433　　Shanghai,200433

电　　话：（021）65904625　　P.R.China

网　　址：http://www.shufe.edu.cn　　http://www.shufe.edu.cn