

高频交易下的股票涨跌预测

梅智敏

院（系）： 计算学部

专 业： 软件工程

学 号： 1183710118

指导教师： 张彦航

2022 年 5 月

哈爾濱工業大學

畢業設計（論文）

題 目 高频交易下的股票涨跌预测

专 业 软件工程

学 号 1183710118

学 生 梅智敏

指 导 教 师 张彦航

答 辯 日 期

摘 要

随着高速计算机的快速发展，高频交易在现代金融交易市场中占据越来越重要的位置。在高频交易的背景下，以限价订单簿 Limit Order Book 为中心的买卖双方匹配机制正被全球大部分股票交易所使用，而 LOB 中的 MidPrice 是一项反应股票市值的重要指标。如何准确预测短期内 MidPrice 涨跌趋势是一大难点与热点，已有的方法主要分为两大类：基于统计模型和基于深度学习模型。前者主要通过对数据或者关键参数的分布做出一定程度的假设来构建模型，但是这种假设却和现代真实的高频数据有很大差异；后者使用数据驱动的方式来学习 LOB 的动态过程，这也正是本论文所主要研究的。本论文首先对于所要进行的预测任务进行了形式化，核心在于将每个时刻市场中的 LOB 状态抽象成向量。随后通过广泛的文献调研和大量的实验验证，本论文设计了两版预测模型：初版模型与终版模型。初版模型使用卷积神经网络 CNN 作为特征提取部分，紧接着使用三通道技术让模型从不同角度进行探索，最后使用 LSTM 网络来捕获 LOB 数据之间的时序依赖关系。终版模型从两个角度对初版模型进行了改进：从模型角度引入 seq2seq 架构让模型可以同时输出多个预测范围 K 下的预测结果，同时使用 cross-attention 技术综合考虑 LSTM 编码器在每一个时间步的输出，获得比初版模型更多的信息；从特征角度引入两个新特征 VOI 和 OIR 以帮助预测，它们可以反应当前市场中买卖双方的压力差值，并且通过自相关测试和与预测标签的相关性测试证明了它们的有效性。此外，为了更好地比较不同模型的性能，本论文还设计了一个基于预测结果的回测框架。此框架以“滑动窗口”的风格来依次读入 100 个时间步的向量化历史 LOB 数据，根据预测模型得到未来 K 个时间步内股票价值是上涨、下跌还是大体保持不变，分别映射到“尽力全仓买”、“尽力清仓”和“保持不变”这三种交易动作，最后可得到盈亏情况。本论文设计了预测实验和回测盈亏实验来测试两版模型的性能并在 Linux 云服务器上使用 RTX A4000 GPU 完成。最终结果显示初版模型相较于已有的一些分类器具有更好的性能而终版模型又在初版模型的基础上有了新的提升。初版模型的预测任务 F1 值和回测盈利率分别可以达到 0.77 和 16%；而终版模型的表现分别为 0.83 和接近 25%。

关键词：高频交易；MidPrice 涨跌；深度学习；回测盈亏

Abstract

With the rapid development of high-speed computers, high-frequency trading plays an important role in the modern financial trading market. In the context of high-frequency trading, the buyer-seller matching mechanism centered on the Limit Order Book is being used by most stock exchanges around the world, and the MidPrice in LOB is an important indicator reflecting the stock market value. How to accurately predict the movement trend of MidPrice in the short term is a challenge. Existing methods are mainly divided into two categories, statistical model and DL model. The former usually makes some assumptions about the distribution of data or parameters, but this is not suitable for real high-frequency data. The latter uses data-driven method to learn the dynamic process of LOB, which is the focus in this paper. Firstly, this paper formalizes the prediction task. The core idea lies in abstracting the LOB state into a vector. Then, through extensive literature review and experimental verification, this paper designs two versions of prediction model. The first version uses convolutional neural network as the feature extraction part, then adds the three-channel technology to explore, and finally utilize LSTM to capture the temporal dependency between LOB data. The final version of the model improves from two perspectives. From the model perspective, seq2seq architecture is introduced so that the model can output the prediction results under multiple prediction ranges K . Besides, the cross attention mechanism is used to comprehensively consider the output of the LSTM encoder at each time step. From the feature perspective, two new features VOI and OIR are introduced. They can reflect the pressure difference between buyers and sellers in the market, and their effectiveness is proved by autocorrelation test and correlation test with forecast labels. In order to better compare the performance of different models, this paper also designs a backtesting framework, which reads the vectorized historical LOB data of 100 time steps in turn in the style of 'sliding window', and obtains the MidPrice movement in the next K time steps according to the prediction model, then carries out trading actions 'buy', 'sell' or 'hold', finally obtains PnL. In this paper, the prediction and backtesting experiment are designed to test the performance of models, which are completed on the Linux using RTX A4000 GPU. The final results show that the first version model has better performance than some existing classifiers, and the final version model has a new improvement. The F1 value of the prediction task and the PnL of the first version model can reach 0.77 and 16% respectively while the final model are 0.83 and close to 25%.

Keywords: high frequency trading, MidPrice movement, deep learning, backtesting

目 录

摘 要.....	I
Abstract.....	II
目 录.....	III
一 绪 论.....	- 1 -
1.1 课题背景及研究的目的和意义.....	- 1 -
1.1.1 高频交易与 Limit Order Book.....	- 1 -
1.1.2 研究目的和意义.....	- 4 -
1.2 国内外研究现状.....	- 5 -
1.2.1 基于统计模型的预测方法.....	- 6 -
1.2.2 基于深度学习模型的预测方法.....	- 7 -
1.3 该领域存在的挑战及本文的主要研究内容.....	- 8 -
1.3.1 该领域存在的挑战.....	- 8 -
1.3.2 主要研究内容.....	- 9 -
1.4 章节安排.....	- 10 -
二 问题形式化及数据集描述.....	- 12 -
2.1 研究问题的形式化.....	- 12 -
2.1.1 MidPrice 涨跌预测形式化.....	- 12 -
2.1.2 交易策略形式化.....	- 14 -
2.2 数据集描述.....	- 15 -
2.2.1 来源及包含的股票.....	- 15 -
2.2.2 特征集.....	- 16 -
2.2.3 标签集.....	- 16 -

三	初步模型的设计与实现	- 17 -
3.1	相关技术介绍	- 17 -
3.1.1	CNN	- 17 -
3.1.2	LSTM	- 19 -
3.2	模型设计思路	- 20 -
3.2.1	CNN 层设计	- 21 -
3.2.2	三通道中间层设计	- 22 -
3.2.3	LSTM 层设计	- 22 -
3.3	模型实现	- 23 -
3.4	交易策略	- 27 -
四	改进模型设计与实现	- 29 -
4.1	从模型角度的改进	- 29 -
4.2.1	Volume Order Imblance	- 31 -
4.2.2	Order Imblance Ratio	- 32 -
4.2.3	特征有效性初步测试	- 32 -
五	实验设计与实现	- 36 -
5.1	模型指标实验	- 36 -
5.1.1	实验环境	- 36 -
5.1.2	实验设计	- 37 -
5.1.3	实验结果与分析	- 38 -
5.2	回测盈亏实验	- 43 -
5.2.1	实验设计	- 43 -
5.2.2	实验结果分析	- 45 -

结 论	- 46 -
参考文献	- 48 -
哈尔滨工业大学本科毕业设计（论文）原创性声明	- 51 -
致 谢	- 52 -

一 绪 论

1.1 课题背景及研究的目的和意义

1.1.1 高频交易与 Limit Order Book

高速计算机与深度学习技术的快速发展给股票市场带来了巨大的转变，越来越多的交易者开始使用计算机程序来代替传统的人工选股、决策买卖时机与数量并提交指令。而高频交易就隶属于程序化交易，它的主要特点就是使用秒级甚至毫秒级的市场实时数据来发掘微小的获利时机并在此基础上频繁地提交买卖指令，虽然每笔交易获利不多，但是聚沙成塔同样可以获取高额利润。

为了更好的进行研究，需要了解现代高频交易市场的运作机制。传统的金融市场是以报价驱动为基础而运作的，即交易员与交易商（也称“做市商”）在市场中进行沟通。交易商负责维护金融资产的库存量，公布出价和要价，并被要求按其报价进行交易。交易商的存在提供了市场流动性，但是报价驱动市场的透明度相对较低。而如今，随着电子通信网络和高频交易的快速发展，订单驱动的金融市场成为了主流。它和传统的报价驱动市场最大的区别就在于市场中的交易员可以直接与其他交易员进行交易而没有中间交易商的干预。而订单驱动的金融市场的核心就是以限价订单簿 Limit Order Book 为中心的双重拍卖机制来促进交易，这一机制已经被世界各大交易所普遍运用，比如美国的纽约交易所（NYSE）、纳斯达克和英国的伦敦证券交易所（LSE）等。

在以 Limit Order Book 为中心的金融市场中，提交给交易所的指令分为两大类：包含价格信息和数量信息的 Limit Order，仅包含数量信息的 Market Order。前者会要求该交易的成交价格必须好于或者等于所指定的值，如果当前市场中最好的价格仍然不能满足要求，这个交易指令就会继续等待直到市场中出现了满足

要求的价格为止，所以此类指令具有“执行不确定性”，即他有可能不会被执行，一直在等待状态。后者则要求该交易立刻以当前市场中最好的价格来完成订单，此类订单一定会被执行，但是不能保证成交价格足够好，所以它具有“价格不确定性”。另外，除了提交这两类指令，交易者也可以取消他们之前所提交但是暂时处于等待状态的 Limit Order。

本质上来说，Limit Order Book（也称为 LOB）是当前金融市场行情的一个快照，其基本功能是匹配市场中的买卖双方，它由股票交易所统一管理并呈现给每一位交易者，以帮助交易者做出交易决策。LOB 包含了一组可以代表市场状态及其变化方式的数据且会动态更新，它是当今大多数电子交易所在金融和加密货币市场中使用的机制。图 1 是 LOB 的示意图，可以帮助我们直观了解。

独立统计：每一支股票的 LOB 数据分开统计，每一个 LOB 只反映其对应的那支股票的信息。

买卖两方：Ask 代表卖方，Bid 代表买方。

价格数量对应关系：它表明当前市场中有人愿意以多少价格出售/买入多少数量的股票，以 Ask 方的 20.6 价格为例，它对应的 Volume 值为 2，代表目前市场中有人愿意以“不低于 20.6”的价格出售 2 支股票。需要注意的是，展示在 Limit Order Book 中的价格都是 Limit Order 中所要求的价格，而 Limit Order 是给出某个价格底线，然后要求成交价“好于或者等于该底线价格”。故 Bid 方 20.5 价格对应 Volume 值为 3 的含义就是：当前市场中有人愿意以“不高于 20.5”的价格买入 3 支股票。

买卖双方不同的 Level：仅仅依据交易者的预期价格而无需考虑对应的 Volume 来对他们提交的“交易需求”进行分级，Level-1 是最高级，当市场中的交易发生时，会优先匹配高等级的“交易需求”再到低等级的需求。首先对于 Bid 方，价格越高则所处的等级越高，故 20.5 的价格处于 Level-1，20.4 的价格处于 Level-2。接

着是 Ask 方，价格越低则所处的等级越高，20.6 的价格处于 Level1，20.7 的价格处于 Level2。

LOB 的更新：每当市场中有交易员提交了新交易指令或者取消了之前的某个 Limit Order，LOB 都会更新以呈现出当前市场的真实交易情况给每一位交易者。图 2 和图 3 详细展示了两种情况下的 LOB 更新过程：（1）当有交易者提交了新的交易指令时，LOB 会发生更新，图 2 中展示的是限价为 20.55 的 Limit Ask Order 取代 20.6 成为了新的 Best Ask。注意：若有交易者提交新的 Market Order 时，则会立刻与 LOB 中记录的 Limit Order 发生匹配，也会导致 LOB 的更新。（2）当有交易者取消了尚未执行 Limit Order 也会引起 LOB 的更新，图 3 中展示的是限价为 20.6 的 Limit Ask Order 被它的提交者给取消了，所以 LOB 就会删除这一指令的记录，导致 20.7 从 Level-2 Ask 变成了 Level-1 Ask（Best Ask）。

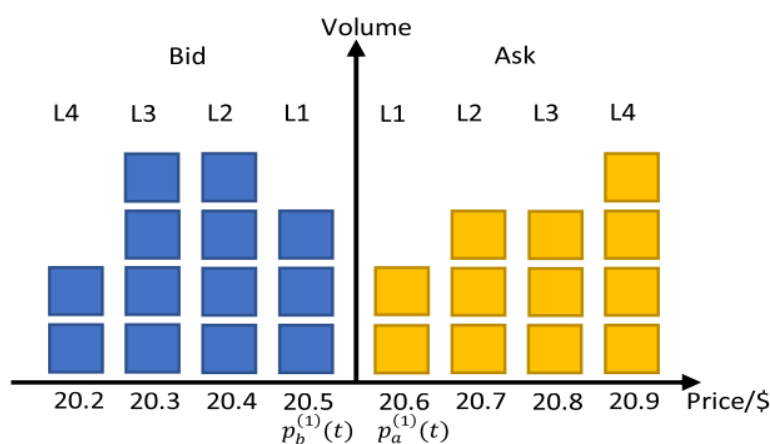


图 1-1 LOB 数据示意图

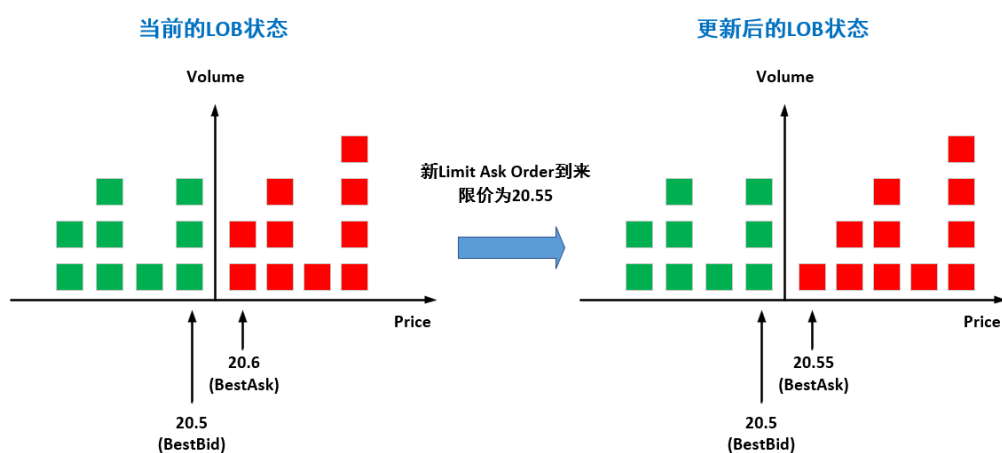


图 1-2 新 Limit Order 到来导致 LOB 更新

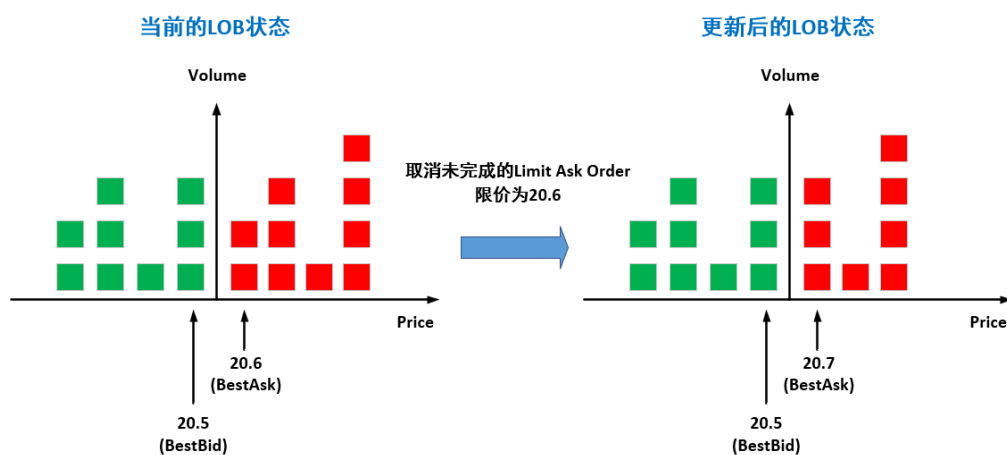


图 1-3 取消未完成的 Limit Order 导致 LOB 更新

1.1.2 研究目的和意义

在基于 LOB 匹配机制的金融市场中，**Best Ask** 为当前市场中针对该股票的卖方最低要价。同理，**Best Bid** 就是买入该股票所能接受的最高价格。在此引入一个新的指标 **Mid Price**:

$$midprice = \frac{BestAsk + BestBid}{2} \quad (1-1)$$

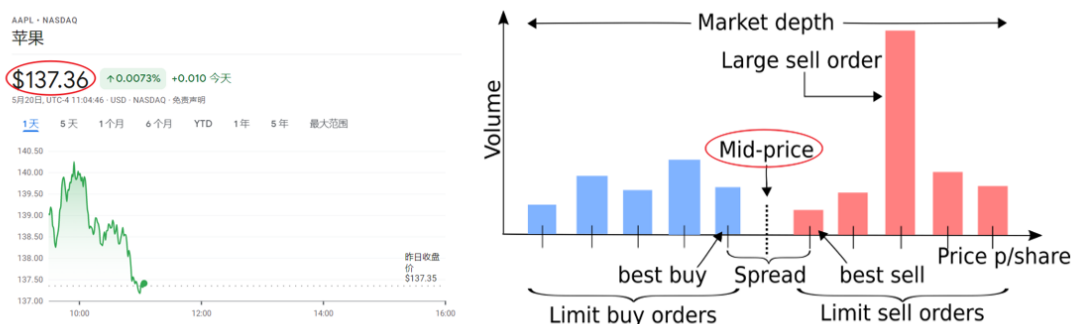


图 1-4 LOB 中 Mid-Price 的意义

在高频交易市场中，Mid Price 是一项重要的指标，它可以表示资产的一般市场价值，所以人们经常使用这一指标来代表资产价值，例如图 4 中苹果 AAPL 股票价格其实就是对应 LOB 中的 Mid Price。

因此，对于量化金融研究者而言，如何准确预测短期内 LOB 中 Mid Price 的涨跌情况成为了近年来的热点问题。目前的高频交易研究主要包含两大类：价格预测和优化策略，而价格预测也正是优化策略的基础。因此借助 LOB 进行短期 Mid Price 变化预测可以有效指导交易者们合理优化自己的交易策略以获得最大收益，具有重要的实际意义。

1.2 国内外研究现状

已有的方法主要可以分为两大类：统计模型方法以及数据驱动的机器学习方法。其中前者是传统的主流方法，优点是不需要利用大量的数据来训练，复现较为简单；缺点是这类方法经常会对数据的分布进行一些和实际情况不相符合的假设，从而导致应用到实际的高随机性的股票数据时，预测效果往往不好。例如^[1]中所提出的 VAR 模型以及^[2]中提出的 ARIMA 模型。而今年来随着深度学习的兴起，研究者们更倾向于使用数据驱动的深度学习方法来完成高频交易下短期股票涨跌趋势预测任务，本课题也挑选了部分已有的方法进行了复现。

1.2.1 基于统计模型的预测方法

试图利用统计模型或者随机模型来捕获 Limit Order Book 中的动态特征，并在此基础上准确预测短期内股票价格涨跌趋势是很多统计学派研究者们所希望做到的。

例如，Gourieroux（1999 年）^[3]、Bouchaud（2002 年）^[4]和 Smith（2003 年）^[5]通过套用多种统计模型进行试验挖掘出了 Limit Order Book 中 20 多种动态特征，并给出广泛统计特征列表。然而，他们的工作仍然不能将特征统一到单个统计模型中加以使用。可喜的是，Cont 和 de Larrard 在 2011 年的工作^[6]中，填补了上述空白，他们提出使用马尔可夫模型来模拟以 Limit Order Book 为基础的金融市场运作过程，该模型捕捉了 Market Order 以及 Limit Order 的主要特征并在此基础上模拟这些特征对价格走势的影响。但是他们的模型仅仅考虑了 LOB 中 Level-1 的特征，即大部分剩余特征并不在考虑范围内。值得一提的是，相同的作者 Cont 在他的另一个工作^[7]中确实考虑到了多个 Level 的 Price 以及对应的 Volume 信息，他假定所有即将到来的市场事件（Limit Order，Market Order 以及 Cancel Order）都遵循独立的泊松过程，以此来预测未来的 MidPrice。受到 Cont 和 de Larrard 使用马尔可夫模型成功的启发，更多基于 Markov 框架进行适应性优化的模型开始涌现，如 VAR^[1]和 ARIMA^[2]。但是此类统计模型完全依赖于手工特征，大多不具备泛用性，它们往往在某几支股票上表现很好但是在其他股票上表现很差。

总的来说，在 2015 年之前，使用传统的统计模型来理解 Limit Order Book 动态过程的研究工作还较多，近年来则是大大减少了。其根本原因就在于这类方法要么大量使用一些人工创建的特征，要么对数据或者市场中事件的分布进行一些和实际情况不相符合的假设，从而导致模型难以应用到实际的高随机性的股票数据。

1.2.2 基于深度学习模型的预测方法

近年来随着深度学习的快速发展以及传统统计模型难以有效捕获 LOB 的动态特征，越来越多的深度学习模型被应用到金融数据中以尝试利用 Limit Order Book 中的多维市场交易信息来预测短期内金融资产（股票、证券以及外汇等）的价格走势。其中 MLP 和 SVM 是两类常用的分类模型，例如在^[8]中，作者训练了一个 SVM 模型来预测未来 30s 内日经 225 指数的价格涨跌走势。在^[9]中，作者使用 12 个维度的特征向量作为输入，来预测韩国综合股价指数的日内价格走势。并且在两个不同大小的预测窗口下（一个为长期，另一个为短期）对 MLP 和 SVM 的性能进行了比较。在^[10]中，作者通过一个预测期货未来价值的任务来对 MLP, SVM 和 RBF-NNs 进行了实验比较。

除了在不同的金融数据集中比较分类器的性能外，也有很多学者将研究重心放在如何从 Limit Order Book 中提取合适的特征来帮助后续预测任务。由于金融数据的高频高随机特性，没有预处理或者特征提取的话会给后续模型训练带来极大困难。为此，^[11]中的作者在模型训练之前使用主成分分析 PCA 来对源数据进行降维从而使得相同的模型性能得到了提升。但是，这种将特征提取方法是静态的预处理步骤，没有随着模型训练过程实时优化以最大限度地提升模型的整体性能。为了弥补这一缺陷，一些学者提出将特征提取功能设计成模型中的某个单独的网络层，这样在模型训练过程中就可以实时利用反向传播来更新自身参数以达到优化的目的。相关的代表工作有^[12]中的 BoF 模型以及^[13]中的 CNN 模型，在^[12]中，作者在公开数据集 FI-2010 上进行了实验，这是基于 Limit Order Book 进行股票价格预测的标准 benchmark 数据集，包含连续 10 天从纳斯达克北欧股票市场中提取的五只股票的 LOB 的标准化数据表示，约 4,000,000 个样本点。在^[13]中，作者使用卷积神经网络 (CNN) 来进行特征提取，创新性地将该方法应用到高频股票数据中。该方法使用来自金融交易所的非公开大规模、高频 LOB 数据

作为输入来预测股票的价格走势，并且在实验中与其他不做特征提取的方法（MLP）进行比较。这类工作表明了从大量数据中提取代表性特征对于理解 Limit Order Book 的动态过程具有重要价值。

还有一类模型近年来也在金融数据领域备受关注，那便是 LSTM^[14]。该模型最初是针对 RNN 面对长时依赖会出现消失梯度问题而设计的^[15]，现在已经广泛应用于 NLP 领域，也常常在 seq2seq^[16]模型架构中充当 encoder 或者 decoder 的编码器。近年来越来越多的研究者使用 LSTM 来抓取股票历史数据之间的“时序依赖信息”从而帮助预测 MidPrice 走势。代表工作有^[17]，作者们使用 1000 只股票的 Limit Order Book 数据来测试他们设计的四层 LSTM 模型。实验证明该模型在样本外依旧能保持不错的 F1 值和 Accuracy，这体现了在金融数据中 LSTM 具有发掘潜力。

总的来说，近年来深度学习的研究愈发火热，基于 DL 模型来理解 LOB 动态过程的研究大量涌现。其主要围绕三大主题：如何从 LOB 中提取合适的特征，如何从时间维度捕获 LOB 数据之间的依赖关系以及如何让模型在不同大小的预测范围下仍保持良好的表现。

1.3 该领域存在的挑战及本文的主要研究内容

1.3.1 该领域存在的挑战

综合考虑前面的领域内研究调查，我将该领域内的挑战总结为三点：

(1) 对于统计模型，难以在足够小的参数空间和合理的分布假设之间做到平衡，往往是顾此失彼。

(2) 对于深度学习模型，如何从 LOB 中提取合适的特征，如何从时间维度捕获 LOB 数据之间的依赖关系以及如何让模型在不同大小的预测范围下仍保持良好的表现都是当前面临的难点。

(3) 需要充分的实验来佐证预测模型的优良性能，主要包括两方面的测试：数据集上的模型指标测试以及回测收益测试。前者是利用一些评价指标如 Accuracy、Precision、Recall 以及 F1 值来比较不同模型在同一个数据集上的表现好坏；后者则是依据预测模型的结果设计一个交易策略，然后在历史数据上模拟执行此交易策略，根据最终得到的收益或者损失 Profit&Loss (PnL) 来从另一个角度衡量不同策略的好坏。

1.3.2 主要研究内容

本次毕设的主要研究内容为在高频交易的应用场景下借助 Limit Order Book 数据来对股票的涨跌走势进行预测，同时根据预测结果制定可以获利的交易策略。研究主要分为以下几个阶段：

(1) 首先需要广泛阅读文献，了解目前的研究者们主要采用统计模型还是深度学习模型、提取了 Limit Order Book 中的哪些特征、设计模型的依据、一般在哪些数据集上进行测试。

(2) 其次，需要从已有文献中获得启发，在贴合应用场景的基础上设计出新的分类模型用于预测短期内 MidPrice 的涨跌走势，模型尽量不要对数据的分布做出不合理的假设。

(3) 另外，从两个大方向来优化自己的模型：模型设计与特征提取。模型设计这块可以尝试引入 seq2seq 架构让模型可以同时输出多个预测范围内的价格走势结果；特征提取这块可以先尝试手动设计几个特征，然后利用和标签之间的相关性检验来证明特征的有效性。

(4) 最后，需要进行充分的实验来评估设计的模型，实验分为两大块：评价指标测试和回测盈亏。每一块都包含 3 大阶段：baseline 方法的表现、初步模型的表现以及优化后模型的表现。需要注意的是实验中需要对预测范围 T 取不同值进行多次验证。

1.4 章节安排

本毕设论文正文部分有五章，概述如下：

第 1 章为绪论，主要分为三个模块。首先介绍了课题的应用场景以及现实意义，详细描述了现代以 Limit Order Book 作为匹配机制的金融市场运作方式，指出基于 LOB 的短期 MidPrice 涨跌趋势预测是一大热点；其次对领域内已有的研究方法撰写了文献综述，对基于统计的预测模型和基于 DL 的预测模型均有阐述；最后从前人的方法中得到启发，确定了本课题模型优化方向以及需要的两大类验证实验。

第 2 章为问题形式化及数据集描述，该章分为两节。第一节对本课题研究的问题进行形式化并统一相关符号以便于论文后续表达；第二节对本课题所使用的公开数据集 FI-2010 进行详细描述，包括特征集以及标签集。

第 3 章为初步模型的设计与实现，该章分为四节。第一节对 CNN 以及 LSTM 核心原理进行介绍，本课题的初步模型架构为 CNN+三通道+LSTM；第二节详细描述初步模型的设计思路与模型结构；第三节结合数据集和模型结构给出数据流向图，详细描述了从模型输入到模型输出的向量维度和形状变化情况；第四节基于初步模型设计了一个交易策略，介绍了设计思路与实现方法。

第 4 章为对初步模型的改进思路与实现，该章分为三节。第一节阐述从模型角度的改进，即引入 seq2seq 结构和注意力机制的动机与设计思路；第二节介绍从特征提取角度的改进，即引入两个新特征 OI 和 OIR 的原因、设计思路和有效

性验证；第三节综合前面两节的内容对初步模型加以优化，给出实现细节和优化后的模型。

第 5 章为实验设计与实现，该章分为两节，每一节都包含模型指标测试和回测盈亏两部分实验，模型指标测试会比较模型的 Accuracy、Precision、Recall 和 F1 值而回测盈亏会比较基于不同模型的交易策略在历史股票数据上的盈亏情况。第一节首先介绍实验设计，然后将初步模型和文献综述中提到的一些已有深度学习方法进行比较；第二节将初步模型和改进后的模型进行了比较，具体行文结构与第一节类似。

在最后的结论中，本论文对完成的工作进行了总结：阐述了初步模型和改进后模型的设计思路中的创新点所在；对两阶段实验得到的结果进行了综合分析评价；同时针对高频交易中的股票涨跌预测问题对未来进行展望，提出了几种可能的研究方法。

二 问题形式化及数据集描述

2.1 研究问题的形式化

第一章中已经对本课题所要研究的问题进行了简略描述，此处给出更为准确的形式化表达。分为两个阶段，第一阶段是预测 MidPrice 的涨跌趋势，第二阶段是基于涨跌预测的交易策略。

2.1.1 MidPrice 涨跌预测形式化

根据第一章中对 LOB 的详细描述，我们可以将 t 时刻的 LOB 状态用向量 x_t 表示：

$$x_t = [p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)]_{i=1}^{depth} \quad (2-1)$$

其中不同的 i 代表 LOB 中不同的 Level，每一个 Level 包含买方价格、买方价格对应的量、卖方价格、卖方价格对应的量这 4 个维度的信息，分别用 $p_a^{(i)}(t)$ 、 $v_a^{(i)}(t)$ 、 $p_b^{(i)}(t)$ 、 $v_b^{(i)}(t)$ 表示，示意图如下：

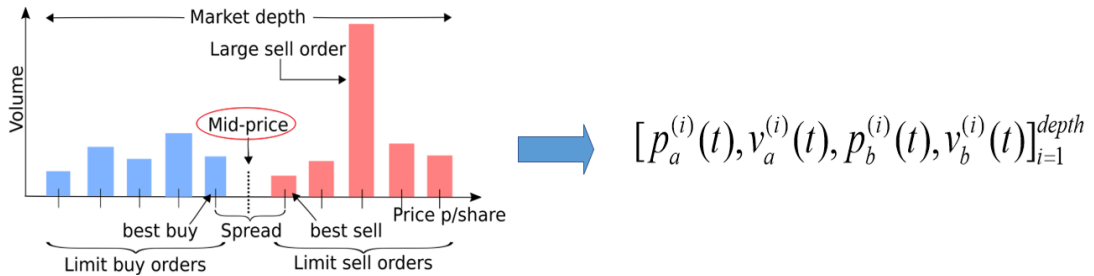


图 2-1 LOB 状态表示

每当市场中交易者提交新 Limit Order、提交新 Market Order 或者取消尚未执行的 Limit Order，作为市场买卖行情快照的 LOB 就会自动更新一次。关注每次更新的 LOB 状态，并用相同的向量化表示它们，就可以得到一串 LOB 状态序列 $[x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, x_{t+3}, \dots]$ 。每一个 x_t 都对应着一个 MidPrice，记作 p_t ：

$$p_t = \frac{p_a^{(1)}(t) + p_b^{(1)}(t)}{2} \quad (2-2)$$

根据第一章的描述， p_t 可以衡量该时刻的股票价值，那么我们就可以通过观察 MidPrice 序列 $[p_{t-2}, p_{t-1}, p_t, p_{t+1}, p_{t+2}, p_{t+3}, \dots]$ 的变化情况来判断股票价值的涨跌。但是由于股票数据具有很强的随机性，如果我们仅仅比较 p_t 和 p_{t+1} 来判断股票价值的涨跌，那么最终得到的结果将会同样随机而难以找到规律。因此，在很多已有的研究中^{[12][18]}，人们引入了一定时间跨度内的增长率来表征股票价值的涨跌情况，具体公式如下：

$$m_+(t) = \frac{1}{k} \sum_{i=1}^k p_{t+i} \quad (2-3)$$

$$m_-(t) = \frac{1}{k+1} \sum_{i=0}^k p_{t-i} \quad (2-4)$$

其中 $m_+(t)$ 代表未来 k 个样本点的平均 MidPrice，而 $m_-(t)$ 代表过去 k 个样本点以及当前样本点的平均 MidPrice，接下来引入增长率的概念：

$$r(t) = \frac{m_+(t) - m_-(t)}{m_-(t)} \quad (2-5)$$

$r(t)$ 表示未来时间范围 k 内的股票平均价值相较于过去时间范围 k 内的股票平均价值涨幅。另外，股票价值可以分为涨、跌、保持相对平稳这三大类，于是可以自定义标签 $y(t)$ 如下：

$$y(t) = \begin{cases} 1 & r(t) > \alpha \\ 0 & \text{其他} \\ -1 & r(t) < -\alpha \end{cases} \quad (2-6)$$

其中， α 是判断阈值，一般取 0.001 到 0.002 之间。在本课题中，沿用前人在^[13]中的取值 0.002。

前人的研究^[17]发现可以使用 LSTM 来捕获 $[p_{t-2}, p_{t-1}, p_t, p_{t+1}, p_{t+2}, p_{t+3}, \dots]$ 之间的时序依赖关系从而帮助预测 $y(t)$ ，我们可以沿用这一思路，但是观察对象是能

提供更多信息的 LOB 状态序列 $[x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, x_{t+3}, \dots]$ 。于是我们将这个预测问题最终形式化如下图：

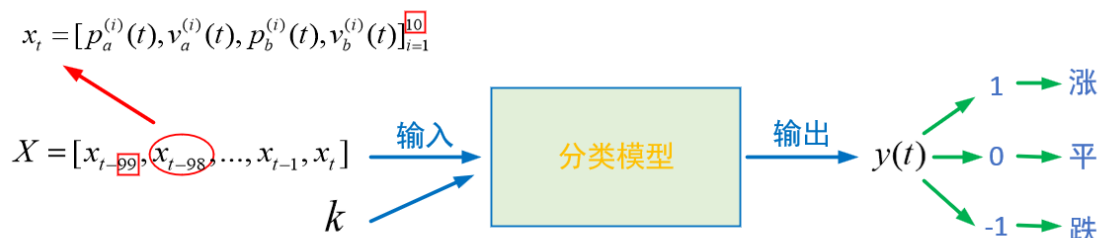


图 2-2 预测问题形式化

我们观察包含当前在内的共计 100 个 x_t 序列，为了控制输入向量的维度，每个 x_t 使用前 10 个 Level 的信息，即每个 x_t 包含 40 个维度的特征。分类模型的输入 X 的维度是 100×40 ，另外需要输入预测范围 k ，经过分类模型处理之后输出对 $y(t)$ 的预测结果：涨或跌或平。

2.1.2 交易策略形式化

依据分类模型的预测结果可以设计出一种简易的交易策略：若 $y(t)=1$ 意味着短期内股票价格上涨，我们需要尽可能买入；若 $y(t)=-1$ 意味着短期内股票价格下跌，我们需要尽可能卖出；若 $y(t)=0$ 意味着短期内股票价格保持平稳，我们就不做任何操作。由此可以定义动作 A_t ，可以取值 *buy*、*sell* 或者 *hold*，形式化表达如下图。假设预测模型在 x_t 处的输出 $y(t)=1$ ，则 $A_t = \text{buy}$ ，我们需要向交易所提交 Market Buy Order 以尽快满仓买入。为了方便后续的回测盈亏实验，我们直接以 BestAsk 作为成交价；同时考虑到模型输出和提交指令的时间耗费，当我们的 Market Buy Order 到达交易所时，LOB 可能已经更新了几次，所以成交价采用 $p_a^1(t + \text{lag})$ 而不是 $p_a^1(t)$ ， lag 为延迟因子，在本课题中取 $\text{lag} = 2$ 。同理，若预测模型在 x_t 处的输出 $y(t)=-1$ ，则 $A_t = \text{sell}$ ，我们需要向交易所提交 Market Sell Order 以尽快空仓，成交价使用 $p_b^1(t' + \text{lag})$ 。

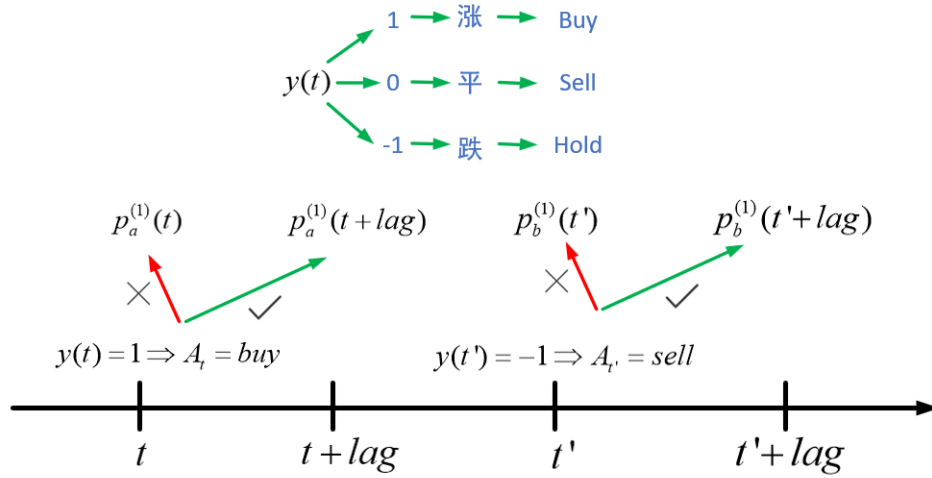


图 2-3 交易策略形式化

2.2 数据集描述

2.2.1 来源及包含的股票

本课题使用数据集是一个公开数据集，名为 FI-2010。它来源于纳斯达克北欧金融市场，共包含五只股票在连续 10 个交易日内的大约 400 万个市场行情快照，即 400 万个 LOB 状态点 x_t ，这些 x_t 按照更新的时间顺序排列且五只股票分开记录。

具体来说，该数据集选用的都是五支股票在芬兰 Helsinki 交易所从 2010 年 6 月 1 号到 2010 年 6 月 14 号期间共 10 个交易日的 LOB 状态信息。数据集通过只选择一个股票市场交易所来避免了与分散市场相关的问题，根据一些学者的研究^[19]，在分散市场的情况下，给定金融资产的 Limit Order 分布在多个交易所之间，这会给后续数据分析带来各种问题。在 Helsinki 交易所，交易时间为 10:00-18:25，中午不休市。但是刚开盘的一段时间和即将收盘的一段时间内，数据波动性极大，因此，为了防止这 2 个特殊时间段内的数据对模型造成影响，该数据集仅保留 10:30-18:00 之间的所有 LOB 状态点。所选用的五支股票详细信息如下：

表 2-1 选用的五支股票信息

ID	ISIN code	Company	Sector	Industry
KESBV	FI0009000202	Kesko Oyj	Consumer Defensive	Grocery Stores
OUT1V	FI0009002422	Outokumpu Oyj	Basic Materials	Steel
SAMPO	FI0009003305	Sampo Oyj	Financial Services	Insurance
RTRKS	FI0009003552	Rautaruukki Oyj	Basic Materials	Steel
WRT1V	FI0009000727	Wärtsilä Oyj	Industrials	Diversified Industrials

2.2.2 特征集

2.1 节中已经对 LOB 状态进行了形式化 $x_t = [p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)]_{i=1}^{depth}$ ，此数据集则对采用更多的特征来描述 LOB 状态，共 144 个特征，可分为 3 大类 9 小类。第一类是 LOB 中的基本特征，和 2.1 节的定义完全一致，共包含 10 个 Level 的信息，即 40 个特征。第二类是由基本特征进行简单四则运算得到的一些进阶特征，且不考虑 x_t 和 x_{t-1} 的时序关系，此类特征共 $20+20+4+2=46$ 个。第三类是考虑了时序关系之后的进阶特征，此类特征共 $40+6+6+6=58$ 个。具体细节如下表：

表 2-2 数据集的特征集合

Feature set	Description	Details
Basic	$u_1 = \{p_i^{ask}, V_i^{ask}, p_i^{bid}, V_i^{bid}\}_{i=1}^n$	10(= n)-level LOB data
Time-insensitive	$u_2 = \{(p_i^{ask} - p_i^{bid}), (p_i^{ask} + p_i^{bid})/2\}_{i=1}^n$	Spread & Mid-price
	$u_3 = \{p_n^{ask} - p_1^{ask}, p_1^{bid} - p_n^{bid}, p_{i+1}^{ask} - p_i^{ask} , p_{i+1}^{bid} - p_i^{bid} \}_{i=1}^n$	Price differences
	$u_4 = \left\{ \frac{1}{n} \sum_{i=1}^n p_i^{ask}, \frac{1}{n} \sum_{i=1}^n p_i^{bid}, \frac{1}{n} \sum_{i=1}^n V_i^{ask}, \frac{1}{n} \sum_{i=1}^n V_i^{bid} \right\}$	Price & Volume means
	$u_5 = \left\{ \sum_{i=1}^n (p_i^{ask} - p_i^{bid}), \sum_{i=1}^n (V_i^{ask} - V_i^{bid}) \right\}$	Accumulated differences
Time-sensitive	$u_6 = \{dp_i^{ask}/dt, dp_i^{bid}/dt, dV_i^{ask}/dt, dV_i^{bid}/dt\}_{i=1}^n$	Price & Volume derivation
	$u_7 = \{\lambda_{\Delta t}^1, \lambda_{\Delta t}^2, \lambda_{\Delta t}^3, \lambda_{\Delta t}^4, \lambda_{\Delta t}^5, \lambda_{\Delta t}^6\}$	Average intensity per type
	$u_8 = \{\mathbf{1}_{\lambda_{\Delta t}^1 > \lambda_{\Delta t}^1}, \mathbf{1}_{\lambda_{\Delta t}^2 > \lambda_{\Delta t}^2}, \mathbf{1}_{\lambda_{\Delta t}^3 > \lambda_{\Delta t}^3}, \mathbf{1}_{\lambda_{\Delta t}^4 > \lambda_{\Delta t}^4}, \mathbf{1}_{\lambda_{\Delta t}^5 > \lambda_{\Delta t}^5}, \mathbf{1}_{\lambda_{\Delta t}^6 > \lambda_{\Delta t}^6}\}$	Relative intensity comparison
	$u_9 = \{d\lambda^1/dt, d\lambda^2/dt, d\lambda^3/dt, d\lambda^4/dt, d\lambda^5/dt, d\lambda^6/dt\}$	Limit activity acceleration

2.2.3 标签集

该数据集自带 5 个标签 $y_{10}, y_{15}, y_{20}, y_{25}, y_{30}$ ， k 代表预测范围，定义同 2.1 节。

三 初步模型的设计与实现

3.1 相关技术介绍

3.1.1 CNN

CNN^[20]最初是应用于图像处理领域，核心功能是挖掘出图像中的隐含特征。对于常见的彩色图像，它向量维度为 $channel \times width \times height$ 。其用于手写数字识别的流程图如下：

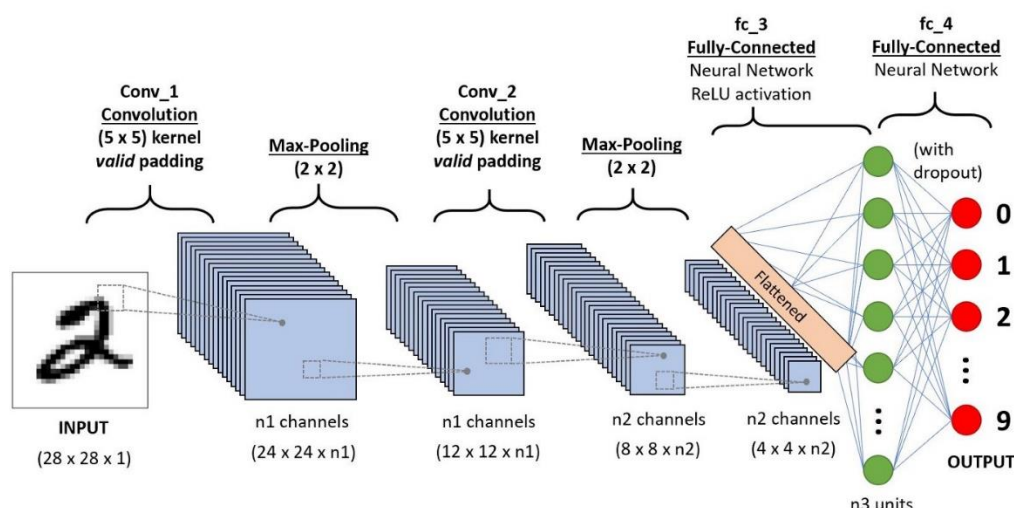


图 3-1 CNN 用于手写数字识别的示意图

CNN 可以接收输入图像，将可学习的 **weights** 和 **bias** 分配给图像中的各个像素，在此基础上让模型可以识别出一些特征。经过足够多次的训练，它有能力将过滤器 **Filter** 中的参数调整到较优的状态，从而更准确地捕获图像中的特征。

CNN 的设计灵感来源与人脑中的视觉皮层组织，单个神经元仅仅对于有限范围内的刺激做出反应。因此，CNN 的核心就在于“过滤器 **Filter**”和“感受区域 **Receptive Field**”的设计。“过滤器”用于捕获图像中的某种特征，“感受区域”就是一块由人工设定好的范围，例如 3×3 ， 5×5 等。“过滤器”每次与图像中一块“感受区域”进行卷积运算，得到的结果存入 **Feature Map**，当“过滤器”扫完一

遍图像范围后即完成了一次特征提取任务，在此之后还可以增加池化操作以及全连接层映射以完成分类等任务。

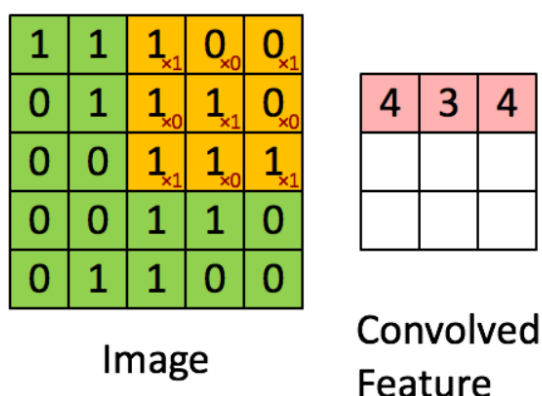


图 3-2 过滤器在感受范围上做卷积运算

上图的绿色部分就是一个 $5 \times 5 \times 1$ 的输入图片，“感受范围”大小为 $3 \times 3 \times 1$ ，黄色部分就是“过滤器”，维度也是 $3 \times 3 \times 1$ ，右下角乘数就是“过滤器”中的权重参数。“过滤器”和“感受范围”做完一次卷积运算后得到结果 4，就是 Convolved Feature 中左上角的数值。

至于池化操作，可以简单理解为对一个大矩阵的信息浓缩从而得到一个小矩阵，而信息浓缩的方式可以自定义，例如 max pooling 就是选择提取池子中最大的数字来代表整个池子的信息而 average pooling 则是使用池子中所有数值的平均数来代表整个池子的信息。具体细节可参考下图：

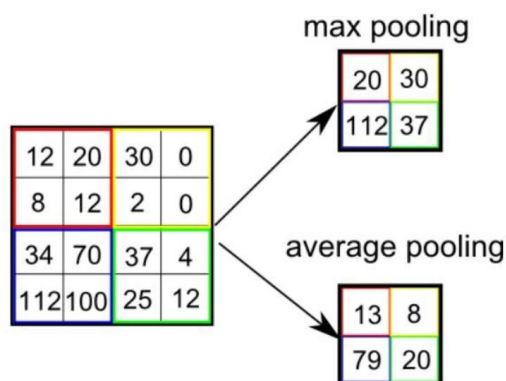


图 3-3 池化操作

值得注意的是，池化操作并不是必要的，随着现在 GPU 算力的增强，许多学者已经在减少池化操作的使用了。总之，经过卷积操作和池化操作之后，模型已经可以更好地理解图像的特征，随后可以将得到的特征向量拉平，再经过全连接层映射到不同的类别上从而完成分类任务。

3.1.2 LSTM

LSTM^[14]起初是为了解决传统 RNN 网络在面对长时依赖时出现的梯度消失问题而设计的改进版 RNN。向后传递 C_t 的巧妙设计使它拥有长时记忆的能力。最初的 RNN 网络被设计为包含简单的循环模块，如下图所示：

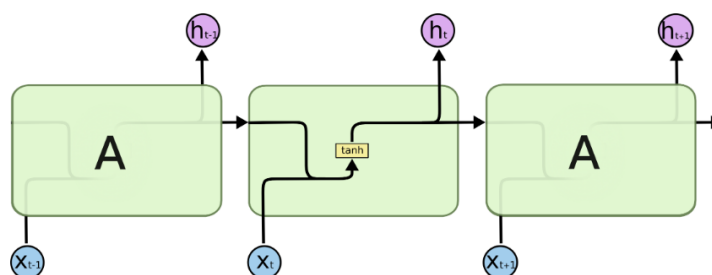


图 3-4 标准 RNN 结构

但是 LSTM 中的重复模块则要复杂地多：

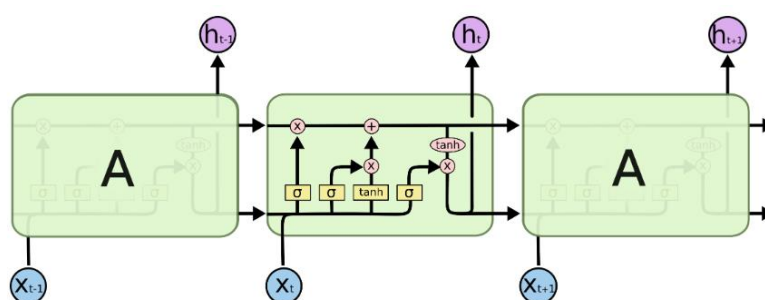


图 3-5 LSTM 结构

将核心的重复模块单独拿出来进行理解，当前时间步输入记作 x_t ，循环向后传递的状态称为 C_t ，最后经过独特的交互得到隐含状态 h_t 并输出，由于 C_t 会沿着重复模块链一直传递下去，所以 LSTM 可以很自然地记住长时依赖关系。

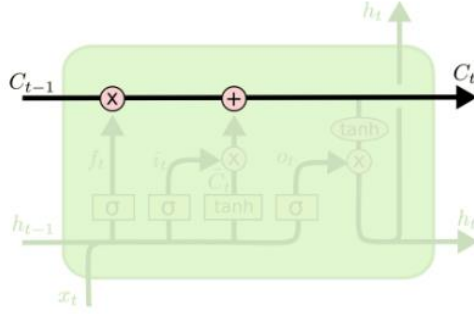


图 3-6 携带的状态

至于如何从 x_t 得到 h_t ，LSTM 利用了巧妙的 gate 设计来决定 gate 前的输入有多少可以保留到 gate 之后，这种选择性地让信息通过的方式可以对模型进行精细调节。首先是 forget gate 的设计，这一步用于决定从携带的 C_t 中丢弃多少信息：

$$f_t = \sigma(W_f \bullet [h_{t-1}, x_t] + b_f) \quad (3-1)$$

然后是 input gate，这一步用于决定往 C_t 中存储哪些新信息：

$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i) \quad (3-2)$$

$$\tilde{C}_t = \tanh(W_c \bullet [h_{t-1}, x_t] + b_c) \quad (3-3)$$

接下来就依据前面 2 个 gate 的输入来更新 C_t ：

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (3-4)$$

最后才是 output gate，它用于决定将 C_t 过滤多少得到最终的输出：

$$\begin{aligned} o_t &= \sigma(W_o \bullet [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh(C_t) \end{aligned} \quad (3-5)$$

总结来说，LSTM 通过巧妙的 gate 设计与 C_t 传递实现了对于序列数据的长期依赖关系挖掘。

3.2 模型设计思路

经过 2.1 节的形式化和 2.2 节的数据集介绍，模型的输入为一个历史 LOB 状态序列 X 和预测范围 k ，模型的输出为 k 对应的 $y(t)$ ，预测短期内 MidPrice 涨跌

情况。考虑前 100 个历史 LOB 状态，每个 LOB 状态仅使用数据集中的基本 40 个特征，故 X 的维度为 (100,40)：

$$X = [x_{t-99}, x_{t-98}, \dots, x_{t-1}, x_t] \quad (3-6)$$

$$x_t = [p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)]_{i=1}^{10} \quad (3-7)$$

从 1.2 节的文献综述和 3.1 节的 CNN、LSTM 技术介绍得到启发：CNN 常常在深度学习模型中用于特征提取以帮助完成后续任务，LSTM 可以帮助挖掘序列数据之间的时序依赖关系。那么在我们的任务中，可以将先用 CNN 提取 X 中的隐藏特征，再经过一些中间层处理之后放入一个 LSTM 网络以抓取不同 x_t 之间的时序依赖，最后再利用全连接层映射到三个类别上。

3.2.1 CNN 层设计

由于 CNN 可以多次叠加使用来达到提取不同粒度范围的特征，因此本课题共使用 3 大卷积层，每一个卷积层的 out_channels 统一设置成 32。

第一卷积层：卷积核 Kernal Size 与步长 Stride 都取 (1, 2)。结合 X 的结构，可以发现使用大小为 (1, 2) 的卷积核可以总结每个 Level 中 (p, v) 对的信息。这里同样需要手动设置步长，因为 CNN 的一个重要特性是参数共享，这一特性很有吸引力，它可以避免学习过多参数导致过拟合。如果仅仅使用默认的 (1, 1)，那就意味着 $(p^{(i)}, v^{(i)})$ 和 $(v^{(i)}, p^{(i+1)})$ 是共享参数的，这显然是错误的，因为价格和数量的动态行为是不同的。

第二卷积层：卷积核大小和步长与第一层保持一致，由于第一卷积层已经捕获了每个 Level 中的独立信息，现在希望第二卷积层可以捕获跨 Level 的信息，设置 (1, 2) 的卷积核与步长正好可以满足这点需求，经过这两层卷积操作之后得到的 Feature Maps 实际上已经包含了 LOB 中的 micro-price 特征，已有的工作^[21]证明这一特征对于预测 MidPrice 有帮助，但与^[21]中不同的是，本课题考虑从 Level-1 到 Level-10 共 10 个 micro-price。Level-1 下的 micro-price 的定义为：

$$p^{micro-price} = Ip_a^{(1)} + (1-I)p_b^{(1)} \quad (3-8)$$

$$I = \frac{v_b^{(1)}}{v_a^{(1)} + v_b^{(1)}} \quad (3-9)$$

第三卷积层：这一层的作用是将特征浓缩，在我们的实验中，经过前面两个卷积层得到的 Feature Maps 维度为（100，10），因此这一层的卷积核大小设置为（1，10），这样就可以将 Feature Maps 维度转变为（100，1）

3.2.2 三通道中间层设计

近年来的深度学习模型中有一种较为流行的 Trick，那就是将同一个输入到多个通道中分别进行处理得到多通道结果，u 它们堆叠到一起再当作后续模型的输入。这个 Trick 的意义就在于多通道结果相当于对于源数据进行了多次探索性特征提取，最后的向量堆叠则是汇总这些探索结果，这大大降低了每个通道进行探索的压力。

受到这种 Trick 的启发，中间层的设计为：将 CNN 层的输出复制 3 份，分别放到 3 个通道中进行探索，最后再进行向量堆叠。前面的 CNN 层的卷积核第一个维度值都是 1，也就是不考虑不同 x_t 之间的时序关系，也没有进行池化操作；而最后的 LSTM 网络可以帮助我们抓取不同 x_t 的长期和短期依赖关系。因此确定三通道中间层的作用如下：

通道一：包含 2 个小卷积层，卷积核大小分别为（1，1）和（3，1），即可以协助捕获 x_t 前后 1 个时间步和 3 个时间步的时序关系，缓解后续 LSTM 层的压力。

通道二：同样包含 2 个小卷积层，卷积核大小分别为（1，1）和（5，1），功能和通道一相似，只不过是另一种探索。

通道三：进行池化操作，池子大小不需要设置太大，本课题使用（3，1），不然可能会丢失一些细粒度的特征。

3.2.3 LSTM 层设计

输入向量经过 CNN 层处理之后分别输入到中间层的三个通道中，处理后的结果进行堆叠再输入到 LSTM 层。本课题的任务本质上是个分类问题，而分类问题最简单的解决办法是使用全连接层，但是这往往会需要学习大量的参数而导致训练过程特别困难。因此 LSTM 的主要作用有 2 个：首先是捕获时序特征，其次是相较于全连接层能够学习更少的参数。

为了捕获提取到的特征中存在的时序依赖关系，将全连接层替换为 LSTM 网络层。LSTM 中每个重复模块所携带的状态 C_t 会反馈给自身，并且传递给下一个重复模块，因此可以对特征的时序动态变化进行建模。在本课题中，我将隐状态 h_t 的维度设置为 64，仅关注最后的 h_t ，将其输入 Fully Connected Layer，输出状态数量为 3，分别映射到涨、跌、平 3 种趋势。最后加上 softmax 层得到 3 种价格变动类别的概率。

3.3 模型实现

根据 3.2 节的详细设计思路，使用 Pytorch 深度学习框架来实现初步模型，模型整体结构图如下：

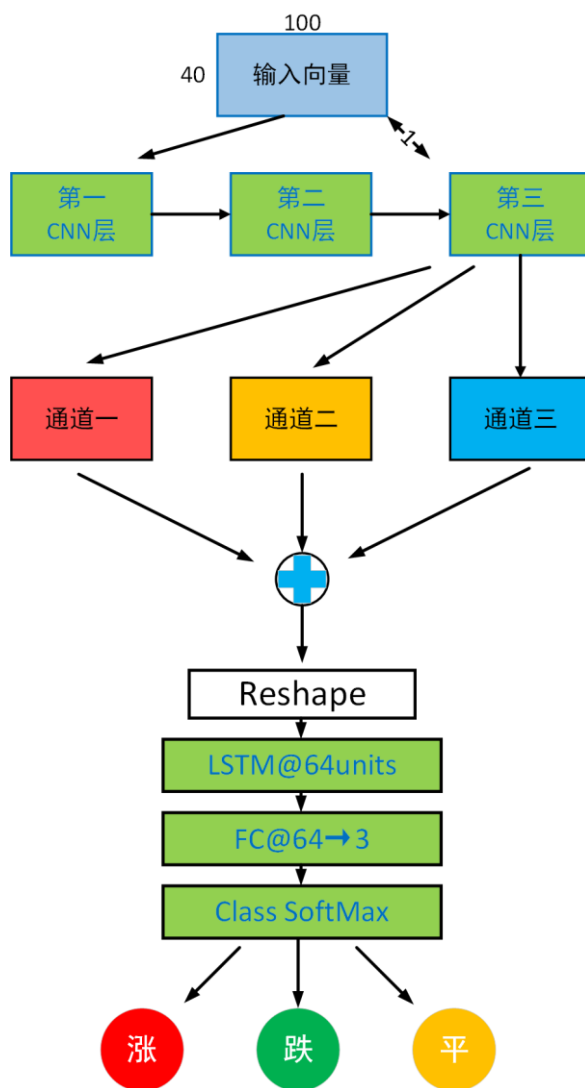


图 3-7 模型整体结构图

输入向量为：

$$X = [x_{t-99}, x_{t-98}, \dots, x_{t-1}, x_t] \quad (3-10)$$

$$x_t = [p_a^{(i)}(t), v_a^{(i)}(t), p_b^{(i)}(t), v_b^{(i)}(t)]_{i=1}^{10} \quad (3-11)$$

由于 Pytorch 中 tensor 的统一形状为 $[BatchSize, Channels, width, height]$ ，示意图隐去了 BatchSize 这一维度，仅仅展示 tensor 的后三个维度，输入向量为 X 拓展了一个维度 $Channels = 1$ 之后的结果。

输入向量首先依次经过三个 CNN 层；然后被复制三份分别进入通道一、通道二和通道三；将三个通道出口处得到的结果 `tensor` 进行 `torch.concat()`，再经过 `reshape` 之后输入到 LSTM 网络中，LSTM 隐状态设置为 64；最后利用 FC 和 SoftMax 层将结果映射到涨、跌、平的概率分布上。

下面详细展示重点网络层的结构，首先是三个 CNN 层：

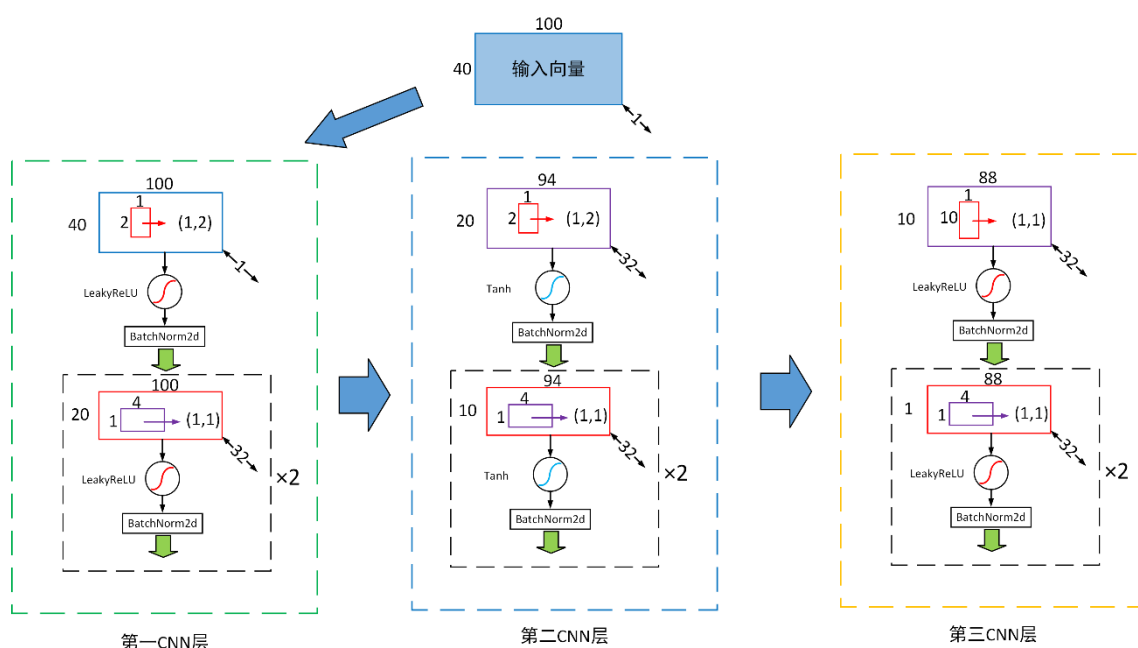


图 3-8 CNN 部分结构图

示意图中矩形的宽、高以及右下角双向箭头中间的数字分别代表 `tensor` 的 `[channels,width,height]`；箭头代表数据流向；大矩形中不同颜色的小矩形代表 Kernel，小矩形右侧的箭头指向的元组代表步长 `Stride`；圆圈加上一段曲线代表激活函数。

第一 CNN 层： 输入向量经过第一次卷积操作，Kernel Size 和 Stride 都取值为 (1, 2)，得到的 Feature Maps 形状为 (32, 100, 20)，紧跟着的是一个激活函数 LeakyReLU 和归一化操作 BatchNorm2d；然后经过两个重复的子网络，即先有一

次卷积，卷积核大小与步长分别为（4，1）和（1，1），然后同样紧跟 LeakyReLU 和 BatchNorm2d。此层的输出 Feature Maps 形状为（32，94，20）。

第二 CNN 层：此层的构造和第一层基本一致，唯一的区别就在于激活函数换成了 Tanh。此层的输入 Feature Maps 形状为（32，88，10）。

第三 CNN 层：此层的构造也和第一层基本一致，唯一的区别在于将第一次卷积操作的卷积核大小与步长切换为（1，10）和（1，1）以便于将特征浓缩。此层的输入 Feature Maps 形状为（32，82，1）。

接下来是三通道结构图：

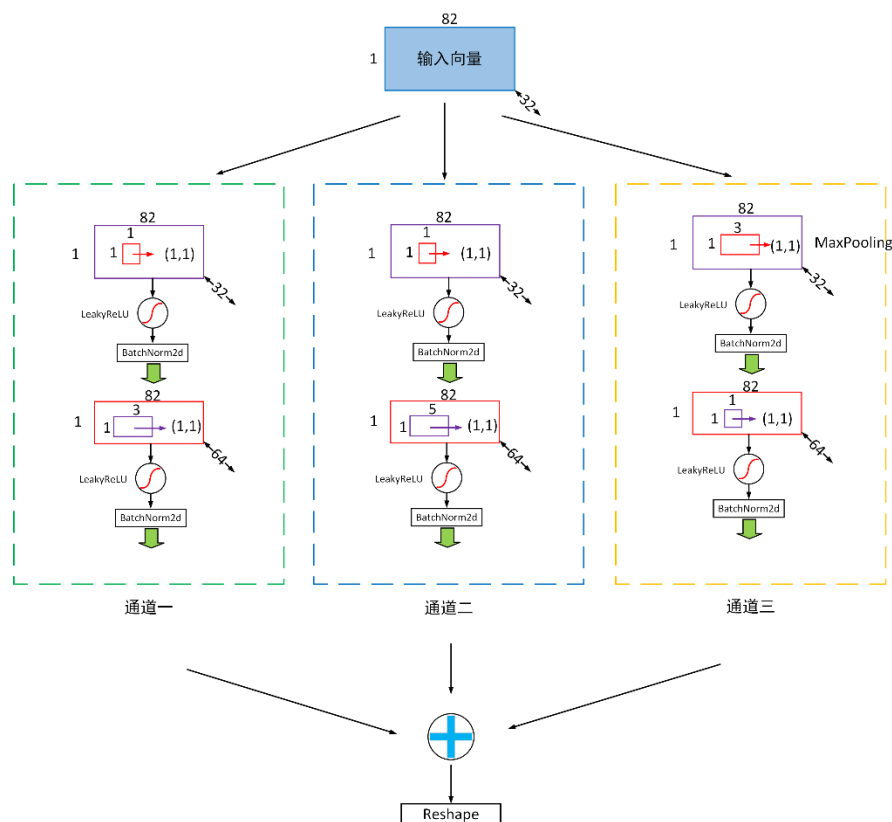


图 3-9 三通道结构图

通道一：CNN 层的输出向量首先经过第一次卷积操作，输出 channels 为 64，Kernal Size 和 Stride 都为（1，1），得到的 Feature Maps 形状为（64，82，1），紧

跟着 LeakyReLU 和 BatchNorm2d；然后经过一次类似流程，仅将 Kernal Size 变为（3，1）。此通道的输出 Feature Maps 形状为（64，82，1）。

通道二：和通道一类似，仅将第二次卷积核大小变为（5，1），此通道的输出 Feature Maps 形状为（64，82，1）。

通道三：和通道一类似，将第一次卷积操作变为池化操作 MaxPooling 并将第二次卷积核大小变为（1，1），此通道的输出 Feature Maps 形状为（64，82，1）。

最后将三个通道的输出进行叠加并 reshape，得到的输出向量形状为（82，192）以符合后续 LSTM 对于输入向量形状的需求。

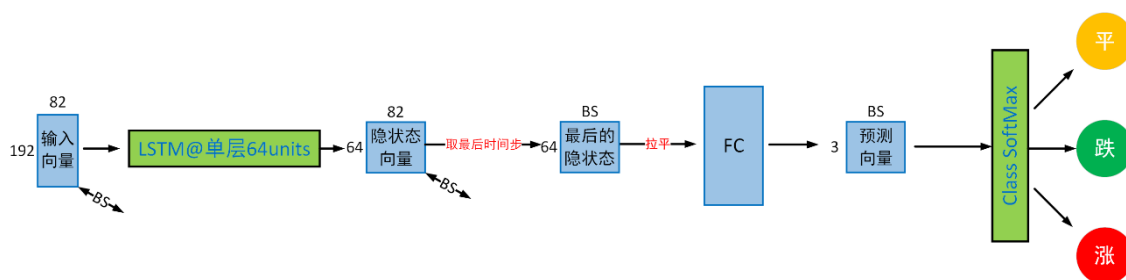


图 3-10 LSTM 部分结构图

上面的 LSTM 部分结构图将 BatchSize 维度展示出来，输入向量的形状就是（BatchSize，82，192）。LSTM 网络设置为单层，隐状态数量设置为 64，即 64 个 LSTM 单元，得到隐状态向量之后取最后一个时间步，得到的向量形状为（BatchSize，64），再经过后续处理得到涨、跌、平的概率分布，取最大概率的趋势作为预测输出。

3.4 交易策略

依据 2.1 节的交易策略形式化以及 3.3 节的预测模型实现，可以设计一个简单的交易策略以供回测盈亏实验使用：

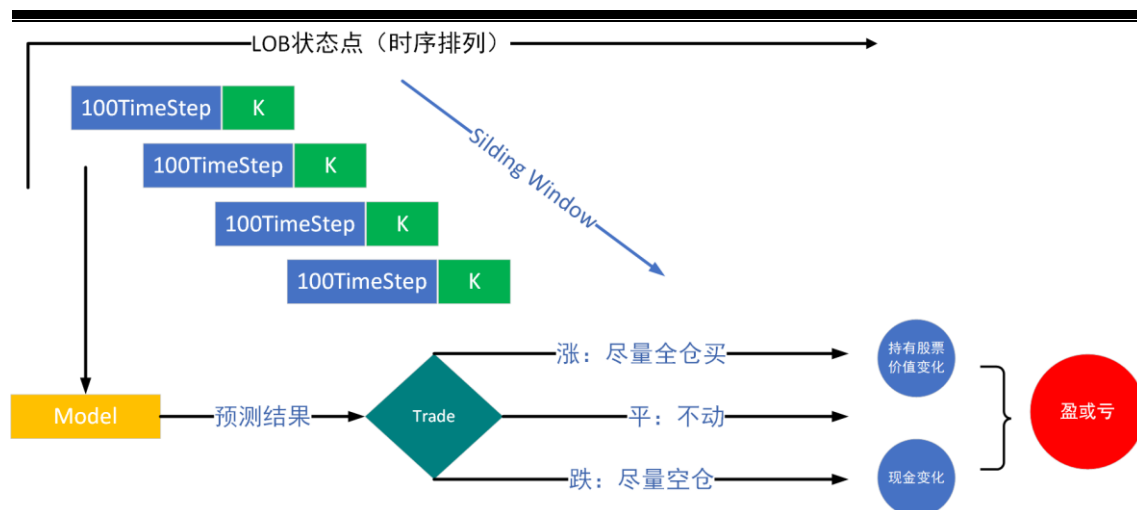


图 3-11 交易策略示意图

如上图所示，交易策略的执行方式类似于“滑动窗口”：读取一段数据，执行相应的动作，然后再次读取下一段数据来执行动作。按照时序读取 LOB 的状态点，每一次都使用过去 100 个时间步的 LOB 状态序列 $X = [x_{t-99}, x_{t-98}, \dots, x_{t-1}, x_t]$ 作为预测模型的输入，预测未来 k 个时间步内 MidPrice 的涨跌。根据预测结果分别做出不同的交易动作，而这些交易动作会引起账号的现金变化以及手中持有股票的价值变化，连续模拟交易一段时间之后便可以得到盈亏情况。

四 改进模型设计与实现

4.1 从模型角度的改进

在初步模型中，需要提前设定好预测范围 K 值，每一个训练好的模型输出对应 K 值下的结果。但是考虑到多预测范围时，初版的模型就稍稍显得低效，因此本节引入了 seq2seq 架构来让模型能够同时输出 $K=10,15,20,25,30$ 时的预测结果；更重要的是，在 seq2seq 架构中，decoder 的前一个时间步的输出可以加入到后一个时间步的输入中从而帮助得到后一个时间步的输出，这是初版模型所做不到的。模型整体架构如下所示：

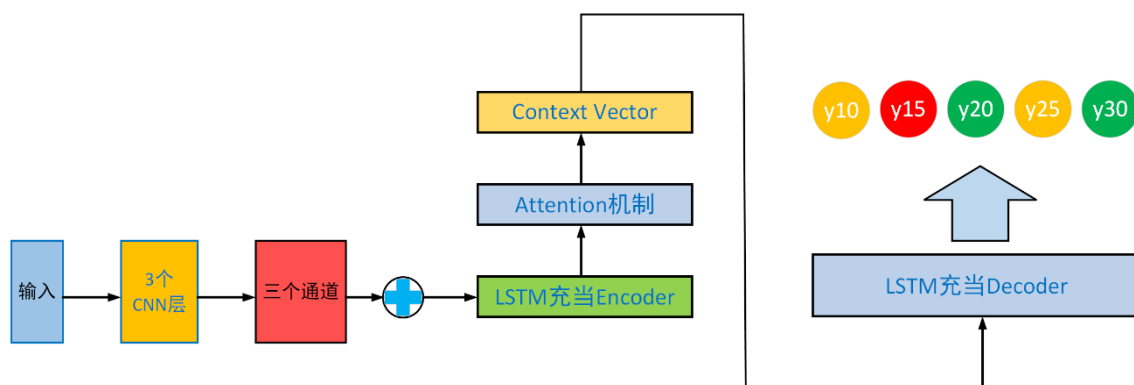


图 4-1 改进后模型整体架构图

主要改动为将原来的 LSTM 层充当 seq2seq 架构中的 Encoder，并且使用每一个时间步的隐状态输出来参与到 Attention 计算，从而得到了考虑整体序列信息的上下文向量 Context Vector。然后同样使用一个 LSTM 网络作为 Decoder 来同时输出 $y_{10}, y_{15}, y_{20}, y_{25}, y_{30}$ ，注意这里需要提前设定好 Decoder 的输出序列长度为 5。当然这里的输出序列长度可以任意选择，只要在训练时给定对应的 y_k 即可。

将 Encoder-Decoder 部分单独拿出来详细解析，结构图如下：

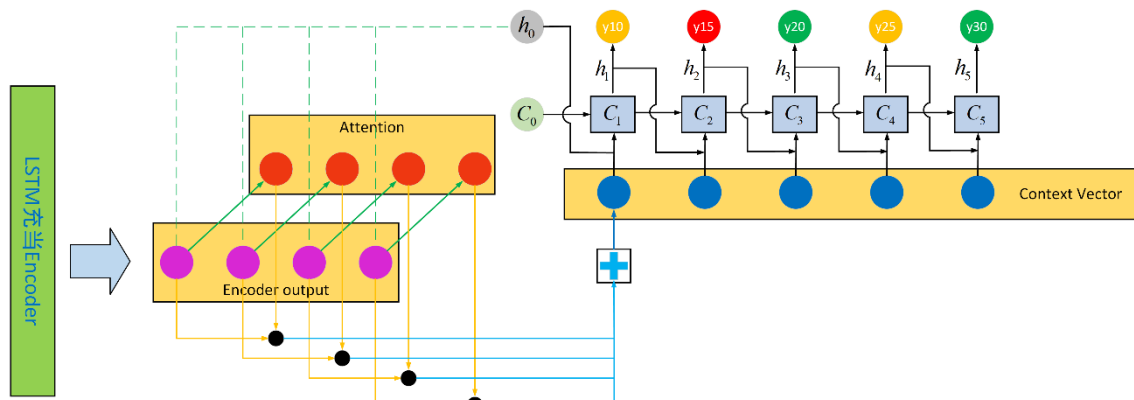


图 4-2 encoder-decoder 详细结构图

初版模型的 LSTM 部分不变，输出得到每一个时间步对应的隐状态，图中使用紫色圆来代表。Decoder 部分使用另一个 LSTM 网络，设定好 Decoder 输出序列长度为 5，并且初始化 h_0, C_0 。首先利用 Decoder 中的 h_0 和 Encoder output 中的每一个结果计算相似性从而得到对应的 Attention 分数，所以这是 cross-attention，图中使用红色圆来表示；然后再将 Attention 和 Encoder output 做点乘得到上下文向量“Context Vector”，图中使用蓝色圆表示；最后进入 Decoder 部分，此处每个时间步的输入包含两部分：Context Vector 与前一个时间步的隐状态；Decoder 输出每个 h_t 之后再经过和初版模型一样的全连接层加上 SoftMax 层可得到不同预测范围下的标签 y_k 。

用公式来描述这个过程，记 Encoder output 为 EO ，Attention 为 A ，上下文向量为 CV 。

Attention 生成:

$$h_t \bullet EO_i = A_i \quad (4-1)$$

Context Vector 生成:

$$\sum_i EO_i \bullet A_i = CV_t \quad (4-2)$$

Decoder 输出:

$$h_t = LSTM(h_{t-1}, CV_t, C_{t-1}) \quad (4-3)$$

得到预测标签：

$$y_k = SoftMax(FC(h_t)) \quad (4-4)$$

其中， h_1, h_2, h_3, h_4, h_5 分别用于产生 $y_{10}, y_{15}, y_{20}, y_{25}, y_{30}$ 。从特征角度的改进

4.2.1 Volume Order Imblance

一些已有的研究^{[22][23]}表明 Volume Order Imblance 对于预测 MidPrice 趋势有帮助，这一指标用于衡量当前市场中买卖双方的不同压力，它的定义如下：

$$VOI_t = \delta V_t^B - \delta V_t^A \quad (4-5)$$

$$\delta V_t^B = \begin{cases} 0 & P_t^B < P_{t-1}^B \\ V_t^B - V_{t-1}^B & P_t^B = P_{t-1}^B \\ V_t^B & P_t^B > P_{t-1}^B \end{cases} \quad (4-6)$$

$$\delta V_t^A = \begin{cases} V_t^A & P_t^A < P_{t-1}^A \\ V_t^A - V_{t-1}^A & P_t^A = P_{t-1}^A \\ 0 & P_t^A > P_{t-1}^A \end{cases} \quad (4-7)$$

其中， $P_t^A, V_t^A, P_t^B, V_t^B$ 分别代表 x_t 中 BestAsk 与其对应的 Volume，BestBid 与其对应的 Volume。

δV_t^B 用来表征 x_t 相较于 x_{t-1} 增加的买方压力：若 $P_t^B < P_{t-1}^B$ 则表明要么有交易者取消了他的 Limit Buy Order 要么是有一个 Order 以 P_{t-1}^B 成交，仅从这两个数据无法判断真实情况到底是属于哪一类，所以将 δV_t^B 设为 0。若 $P_t^B = P_{t-1}^B$ ，就可以使用 $V_t^B - V_{t-1}^B$ 来代表增加的买方压力。最后，若 $P_t^B > P_{t-1}^B$ ，可以将其解释为由于交易者打算以更高的价格购买而导致的价格上涨势头，因此使用 V_t^B 来表征买方压力即可。 δV_t^A 用来表征 x_t 相较于 x_{t-1} 增加的卖方压力，对于它的理解与 δV_t^B 类似。若 $VOI > 0$ 则可以理解为市场中增加的买方压力占据主流，这是一个积极的买入信号，反之，若 $VOI < 0$ 则可以理解为市场中增加的卖方压力占据主流，这是一个积极的卖出信号。

4.2.2 Order Imblance Ratio

前文提出的 VOI 指标仅衡量了 LOB 中不平衡的程度，不足以描述市场中交易员的行为。例如，如果当前 δV_t^B 为 300，当前 δV_t^A 为 200，则 VOI 为 100，这是一个强烈的买入信号。然而，这并没有考虑到 V_t^B 和 V_t^A 之间的比率关系，而这个指标实际上可以反应市场上潜在买家的实力：若 V_t^B 越大则表明买家资金实力越雄厚。因此，定义一个新的指标 Order Imblance Ratio，其定义如下：

$$\rho_t = \frac{V_t^B - V_t^A}{V_t^B + V_t^A} \quad (4-8)$$

分子可以衡量市场中潜在买家的资金实力，分母是为了消除不同时刻 $Volume$ 数量级可能不同带来的影响，起到控制数值范围的作用。该指标越接近 1 则表示买入信号越积极；越接近 -1 则表示卖出信号越积极。这个指标是对于 VOI 的补充，使我们能够识别出 VOI 较大但 ρ_t 较小的情况。在上面的例子中， ρ_t 仅为 0.2，这表明买入信号并没有 $VOI = 100$ 所表征的那么强。

4.2.3 特征有效性初步测试

新增两个特征 VOI, OIR 并将它们使用基于 Z -score 的标准化方法之后加入到 x_t 中，此时 x_t 的特征综述来到了 42 个。对于随机变量 X 的标准化公式如下：

$$X_{std} = \frac{X - \mu}{\sigma} \quad (4-9)$$

其中 μ, σ 分别表示 X 的均值和标准差，记标准化后的特征为 VOI_{std}, OIR_{std} 。随后对 VOI_{std}, OIR_{std} 进行进一步探索，主要使用相关性测试来帮助验证这两个新指标的有效性，这包含两部分：自相关测试以及和标签的相关性测试。使用的相性计算方法为皮尔逊相关性系数：

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \quad (4-10)$$

首先是自相关测试，根据 VOI_{std} 和 OIR_{std} 的定义可知它们本质上仍然是时序数据，因此可以利用针对时序数据的自相关测试来判断它们是否具备“时序稳定性”。对于时序数据而言，若想借用历史数据来预测未来数据则它必须具备一定的程度上的“时序稳定性”，否则的话借用历史数据是没有意义的。

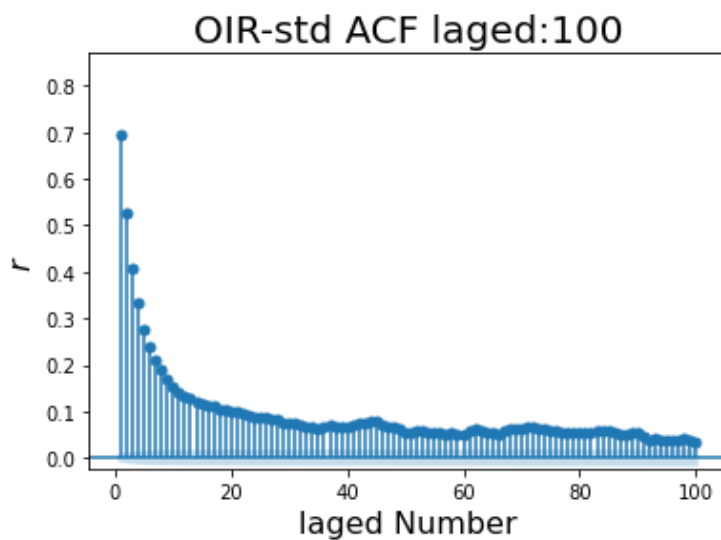


图 4-3 OIR_{std} 的自相关函数图，延迟数为 1 到 100

上图是 OIR_{std} 的自相关函数图， X 轴表示延迟数量， Y 轴表示相关性大小，记延迟数量为 N ，则用于计算相关性的两个序列分别为：

$$[OIR_{std}^{(t)}, OIR_{std}^{(t+1)}, OIR_{std}^{(t+2)}, OIR_{std}^{(t+3)}, OIR_{std}^{(t+4)}, \dots] \quad (4-11)$$

$$[OIR_{std}^{(t-N)}, OIR_{std}^{(t-N+1)}, OIR_{std}^{(t-N+2)}, OIR_{std}^{(t-N+3)}, OIR_{std}^{(t-N+4)}, \dots] \quad (4-12)$$

通过展示延迟数量在 15 之内的 OIR_{std} 自相关函数，可以发现 OIR_{std} 的自相关现象还是比较明显的，且都是正相关关系，尤其是当延迟数为 1 和 2 时，自相关系数分别达到了 0.7 和 0.5 以上，这表明 OIR_{std} 具有良好的“时序稳定性”：

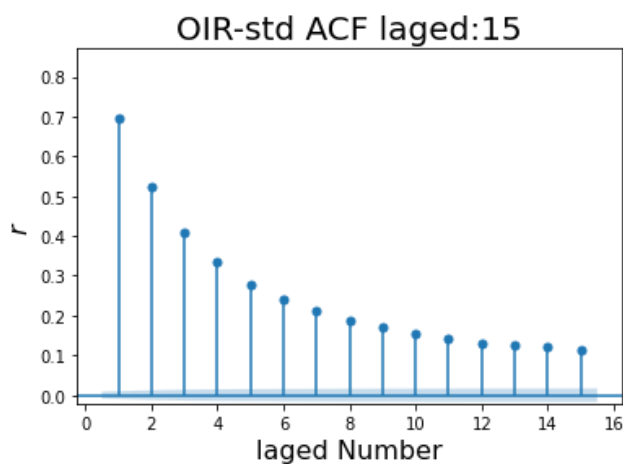


图 4-4 OIR_{std} 的自相关函数图，延迟数为 1 到 15

下面展示一下 VOI_{std} 在延迟范围小于 15 时的自相关函数图，其效果与 OIR_{std} 类似，同样说明其具有一定程度的时序稳定性：

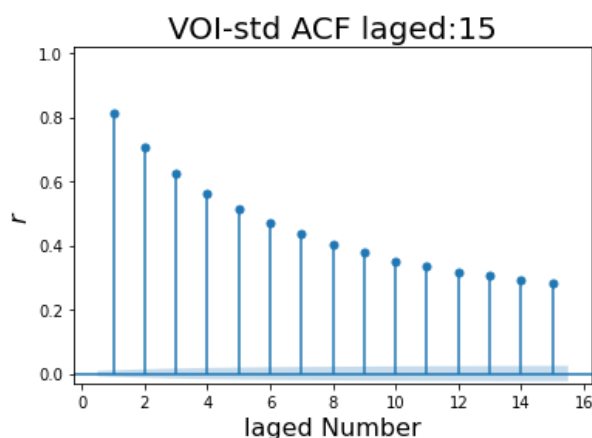


图 4-5 VOI_{std} 的自相关函数图，延迟数为 1 到 15

接着是与标签的相关性测试，目的是为了从直观角度粗略验证 VOI_{std} 和 OIR_{std} 对于预测 MidPrice 涨跌标签是否有帮助。总共针对 5 个标签进行了测试，标签对应的预测范围 K 值分别取 10, 15, 20, 25, 30，共使用 $VOI_{std}, VOI, OIR_{std}, OIR$ 这四个指标，结果统计如下：

表 4-1 4 个指标与 5 个标签之间的相关性测试

	VOI	VOI_{std}	OIR	OIR_{std}
y_{10}	0.1051	0.1203	0.2479	0.2529
y_{15}	0.1055	0.1222	0.2215	0.2327
y_{20}	0.0998	0.1169	0.1968	0.2111
y_{25}	0.0977	0.1140	0.1728	0.1881
y_{30}	0.0943	0.1090	0.1578	0.1727

通过上表可以发现这两个指标对于预测 MidPrice 涨跌标签还是具有一定帮助的，单特征能够和最终的预测标签相关性达到 0.2 左右已经是相当不容易。其中，随着预测范围 K 的增加，四个指标的相关性都有所下降，这也与常规认知相符合，即短期预测的难度一般是要小于长期预测的。另外，横向对比 VOI 与 VOI_{std} ，以及 OIR 与 OIR_{std} 可以发现标准化之后的指标相较于之前具有更强的相关性，这也证实了我们的标准化操作是具有实际意义的。

五 实验设计与实现

5.1 模型指标实验

5.1.1 实验环境

本实验借助 Featurize 云服务器平台实现，相关配置如下表：

表 5-1 服务器硬件配置

属性	值
GPU	RTX A4000×1
显存	16.9GB
CPU	E5-2680 v4×6
内存	30.1GB
硬盘	483.2GB

本实验使用 Anaconda 配置深度学习环境，具体如下表：

表 5-2 代码运行环境

属性	值
系统内核	Linux 5.4.0-91-generic
操作系统	Ubuntu 20.04
CUDA	11.2
Pytorch	1.10
Tensorflow	2.8.0
Keras	2.8.0
Conda	4.10.3
Python	3.7.13

Scikit-learn	1.0.2
Numpy	1.21.5
Pandas	1.3.4
Matplotlib	3.5.1
Statsmodels	0.13.4

5.1.2 实验设计

使用的 FI-2010 数据集共包括 5 支股票在 10 个交易日内的数据，划分训练集、验证集、测试集的比例为 6: 1: 3，且将不同的股票分开使用。这部分实验通过比较 Accuracy、MacroAvg-Precision、MacroAvg-Recall、MacroAvg-F1score、WeightedAvg- Precision、WeightedAvg- Recall 和 WeightedAvg- F1score 共 7 个指标来对比不同模型在预测 LOB 的 MidPrice 涨跌趋势任务中的性能。以本实验的三分类任务为例，解释 MacroAvg 以及 WeightAvg 的含义：针对“涨”、“跌”、“平”这三个类别，首先分别计算得到每个类别各自的 Precision 和 Recall，公式如下：

$$Precision = \frac{TP}{TP + FP} \quad (5-1)$$

$$Recall = \frac{TP}{TP + FN} \quad (5-2)$$

假设“涨”、“跌”、“平”这三类的真实标签数量比值为 $a:b:c$ ，则有：

$$MacroAvg_Precision = \frac{Precision_1 + Precision_2 + Precision_3}{3} \quad (5-3)$$

$$WeightedAvg_Precision = \frac{a * Precision_1 + b * Precision_2 + c * Precision_3}{a + b + c} \quad (5-4)$$

其中 $Precision_1, Precision_2, Precision_3$ 分别为三个类别各自的 Precision。

MacroAvg 就是不考虑类别数量的影响而 WeightedAvg 会使用类别数量作为权重来加权。

针对不同的预测范围 K ，共进行 5 组实验，分别取 $K=10, 15, 20, 25, 30$ 。至于模型的比较部分，选用的模型有：线性分类器 Linear Classifier (LC)、多层感知机 MLP、CNN、LSTM、初版模型和改进后模型。

5.1.3 实验结果与分析

预测实验分别针对五支股票训练模型，最终取平均值进行展示，下面是不同预测范围 K 下得到的实验结果表：

表 5-3 $K=10$ 时模型对比结果

K=10	Accuracy	M_Precision	M_Recall	M_F1	W_Precision	W_Recall	W_F1
LC	72.35	24.12	33.33	27.99	52.34	72.35	60.74
MLP	61.47	20.49	33.33	25.38	37.79	61.47	46.81
CNN	75.14	44.34	36.89	36.62	66.46	75.14	68.88
LSTM	76.37	61.53	54.48	57.02	73.77	76.37	74.51
初版	78.91	78.47	78.91	77.66	75.66	78.41	76.64
终版	83.33	80.53	66.79	71.73	82.75	83.33	81.98

表 5-4 $K=15$ 时模型对比结果

K=15	Accuracy	M_Precision	M_Recall	M_F1	W_Precision	W_Recall	W_F1
LC	63.53	21.18	33.33	25.90	40.36	63.53	49.37
MLP	48.93	46.21	44.28	40.71	50.72	48.93	46.43
CNN	60.55	33.77	33.90	33.02	54.55	60.55	56.96

LSTM	63.89	50.28	43.46	42.95	59.46	63.89	58.84
初版	78.01	72.03	74.04	72.84	71.24	70.97	69.77
终版	75.15	71.61	62.15	65.30	74.11	75.15	73.46

表 5-5 K=20 时模型对比结果

K=20	Accuracy	M_Precision	M_Recall	M_F1	W_Precision	W_Recall	W_F1
LC	57.76	19.25	33.33	24.41	33.36	57.76	42.30
MLP	44.85	44.61	45.41	42.99	47.10	44.85	44.19
CNN	55.15	40.62	40.98	40.43	56.77	55.15	55.76
LSTM	69.00	66.21	62.37	63.26	69.54	69.00	68.65
初版	74.85	74.06	74.85	72.83	73.22	72.89	72.41
终版	77.35	75.79	70.28	72.39	76.94	77.35	76.59

表 5-6 K=25 时模型对比结果

K=25	Accuracy	M_Precision	M_Recall	M_F1	W_Precision	W_Recall	W_F1
LC	49.03	61.76	33.64	22.83	59.75	49.03	32.84
MLP	40.77	45.69	42.02	35.82	45.90	40.77	35.19
CNN	41.11	43.61	43.59	36.95	57.87	41.11	43.16
LSTM	66.67	65.15	62.15	63.17	66.24	66.67	65.97
初版	75.01	75.01	75.01	74.69	74.16	74.87	73.68
终版	79.37	78.64	76.82	77.58	79.20	79.37	79.13

表 5-7 K=30 时模型对比结果

K=30	Accuracy	M_Precision	M_Recall	M_F1	W_Precision	W_Recall	W_F1
------	----------	-------------	----------	------	-------------	----------	------

LC	37.68	47.75	39.67	31.22	49.01	37.68	29.92
MLP	50.67	33.75	43.07	37.83	39.80	50.67	44.57
CNN	44.55	45.60	46.27	43.82	50.72	44.55	45.53
LSTM	56.67	62.62	56.10	55.54	62.81	56.67	55.76
初版	74.34	75.03	74.44	74.40	75.29	74.34	74.48
终版	80.69	70.74	80.66	80.66	80.86	80.69	80.74

表格中每个指标的最大值用红色标注，第二大的值用蓝色标注。更直观地，展示 K=30 时不同预测器的预测热力图：

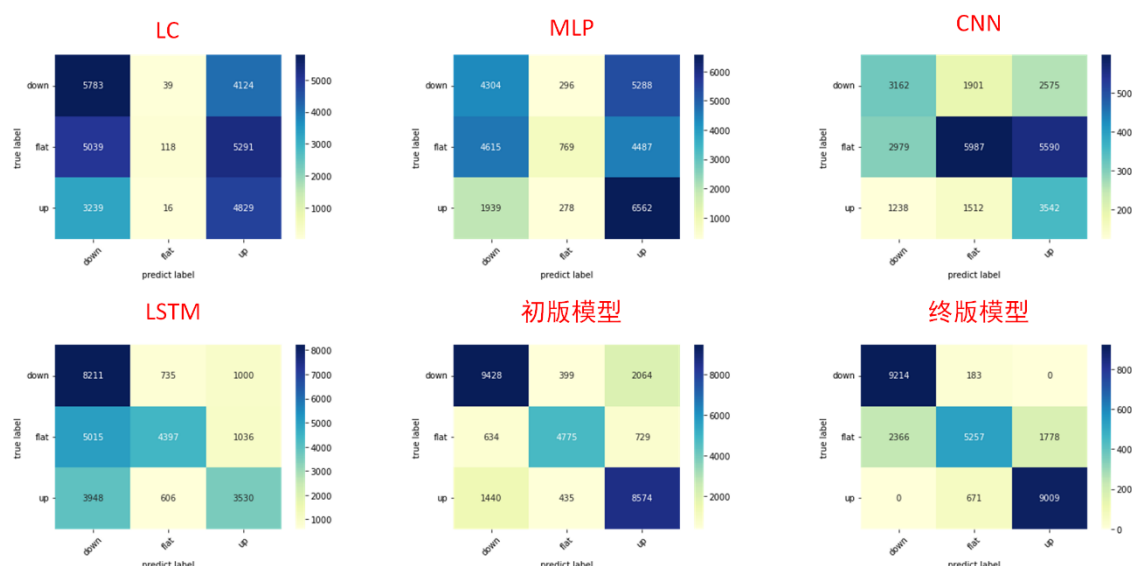


图 5-1 K=30 时各个模型的预测分类热力图

通过观察可以发现：综合来看，初版模型的表现要好于其他几个 baseline 分类器；同时终版模型又在初版模型的基础上有了一定程度的提升。下面从三个角度细致来看：

不同指标的角度：

共 7 个指标，主要可分为 3 组：Accuracy 为第一组，MacroAvg 中的三个指标为第二组，WeightedAvg 中的指标为第三组。可以发现有些模型的 Accuracy 会

很高但是 MacroAvg 却很低，例如 K=10 时的线性分类器 LC，它的 Accuracy 来到了 72.35，接近初版模型的 78.91，但是 LC 的 MacroAvg-F1 值仅有 27.99，远远低于初版模型的 77.66。这种 Accuracy 虚高的情况会让我们误以为这些模型表现很好，但是通过观察该预测模型的分​​类热力图可知它将所有的样本都预测为“保持不变”，而真实标签中该类所占比例为 58%，这就导致 Accuracy 肯定不会低。

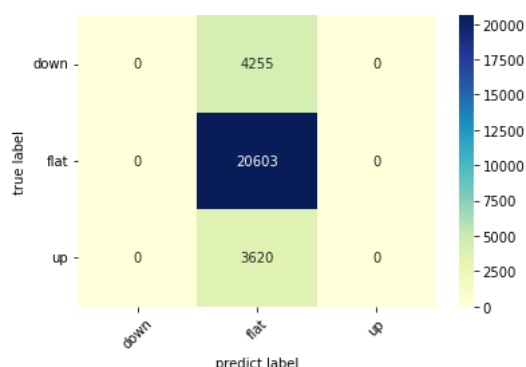


图 5-2 K=10 时 LC 分类热力图

所以我们在评估模型时不可以只看 Accuracy，另外，第三组的指标会将真实标签中的类别占比用来加权，这一般来说是有效的，但是在这个例子中，同样由于“保持不变”占比过大导致该组指标虚高。所以，实验结果证明了本论文选用三组不同指标的合理性。

	precision	recall	f1-score	support
0	0.0000	0.0000	0.0000	4255
1	0.7235	1.0000	0.8396	20603
2	0.0000	0.0000	0.0000	3620
accuracy			0.7235	28478
macro avg	0.2412	0.3333	0.2799	28478
weighted avg	0.5234	0.7235	0.6074	28478

图 5-3 K=10 时 LC 分类报告

不同 K 的角度：

在普通的时序数据预测任务中，对于同一个模型来说，随着预测范围 K 的增大，模型表现应该是逐渐下降的。但是在此任务中却发现了不一样的规律：随着

K 的增大，各个模型的表现总体上是先下降后上升。例如终版模型在 $K=10$ 时 Accuracy、MacroAvg-F1、WeightedAvg-F1 分别为 83.33、71.73、81.98，整体表现最优；当 K 增大到 15 时，三个指标都有所下降，分别为 75.15、65.30、73.46；而当 K 增大到 30 时，指标都回暖到了 80 以上。其余的模型例如 CNN 和初版模型都有此类规律。我对于这种现象的解释为：当 K 很小时，前 100 个时间步的数据与目标范围内的数据关联性更强。随着 K 逐渐增大，这种关联性会变得更加弱，所以相同模型的表现会有所下降。与此同时，标签会变得更加“平坦”，即 $y_k^{(t)}$ 会更倾向于和 $y_k^{(t-1)}$ 保持一致，因为如果想要让标签发生“变化”，则需要 $MidPrice_{t+k}$ 突然有突增或者突减，而这会随着 K 的增大变得愈发困难。因此当 K 增加到 30 时，标签会最为平坦，从而帮助我们进行预测，使得模型的表现有所回暖。

不同模型的角度：

通过观察结果可以发现上述 6 个模型总体性能排序为：

$$LC \approx MLP < CNN < LSTM < \text{初版模型} < \text{终版模型}$$

LC 和 MLP 都难以挖掘有效信息，除了部分情况下 Accuracy 虚高，这两个模型的剩余 6 个指标都在 45 左右，很难突破 50。这证明了仅仅使用简单结构的模型并不能完成 LOB 中 MidPrice 预测任务。

CNN 的这些指标在 50 左右浮动，最好的情况下可以到 55，但是仍然不理想。此网络的关注重点在于每个 x_t 的隐含特征而不是 x_t 与 x_{t-1} 之间的时序依赖关系，实验结果证明这种处理方式有一定的帮助但是并不能仅仅依靠 x_t 的隐含特征来完成此预测任务。

LSTM 的表现相较于前面 3 个模型好了很多，指标在 55 到 60 之间浮动，最好的情况下可以接近 70，这证明了挖掘时序依赖对于此任务的必要性，前人的研究^[17]也说明了这点。

初版模型的表现相较于前 4 个 baseline 方法要更好，总体指标可以超过 70，面对如此多变的高频股票交易市场数据实属不易。结果说明将挖掘 x_t 的隐含特征以及不同 x_t 之间的时序依赖关系结合起来是一条可行的道路。

终版模型相较于初版模型又有了提升，尽管 MacroAvg-F1 值和 MacroAvg-Recall 在 $K=10, 15, 20$ 时有所下降，但是当 K 增大到 30 时终版模型实现了反超，WeightedAvg 组的指标平均提升 5 个百分点。这证明了 seq2seq 架构的泛用性以及 VOI、OIR 指标的有效性。seq2seq 架构和 attention 机制在 NLP 领域大行其道，本论文将其引入到量化金融研究中并且取得了成功，这给后续研究道路指明了一种方向，即可以参考一些在其他领域取得成功的模型，然后将其迁移到新场景中并结合一些领域内知识，这或许可以得到不错的结果。

5.2 回测盈亏实验

5.2.1 实验设计

首先设计了一个账户类，包含两个属性：现金和持有的股票数量，同时包含两个简单的方法，分别用于处理买股票和买股票。

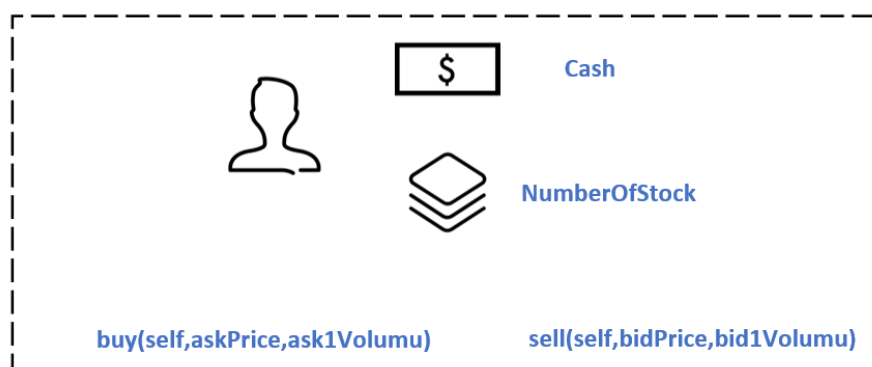


图 5-4 账户类

Buy 方法用于“尽力全仓买”，它的参数 askPrice 选取 $p_a^{(1)}(t + laged)$ ，ask1Volume 就是 $p_a^{(1)}(t + laged)$ 在 LOB 中所对应的 Volume。允许购买的最大股票

数量为，其意义是满足 LOB 中 Volume 约束和资金约束的情况下购买尽可能多的数量：

$$buyNumber = \min(\text{math.floor}(\text{self.cash} / \text{askPrice}), \text{ask1Volumu}) \quad (5-5)$$

类似的，Sell 方法用于“尽力清仓”，它的参数 bidPrice 选取 $p_b^{(1)}(t + laged)$ ，bid1Volume 就是 $p_b^{(1)}(t + laged)$ 在 LOB 中所对应的 Volume。允许卖出的最大股票数量为：

$$sellNumber = \min(\text{bid1Volumu}, \text{self.numberOfStock}) \quad (5-6)$$

回测盈亏实验的框架图如下，按照“滑动窗口”方式来进行，即每一次先读取历史 100 个时间步数据以预测未来 K 时间范围内走势，根据预测结果执行不同买卖动作；接着“窗口”向右滑动一段距离，重复同样的流程。其中“窗口”每次滑动的距离称为“窗口大小”，本实验中取值为 5：

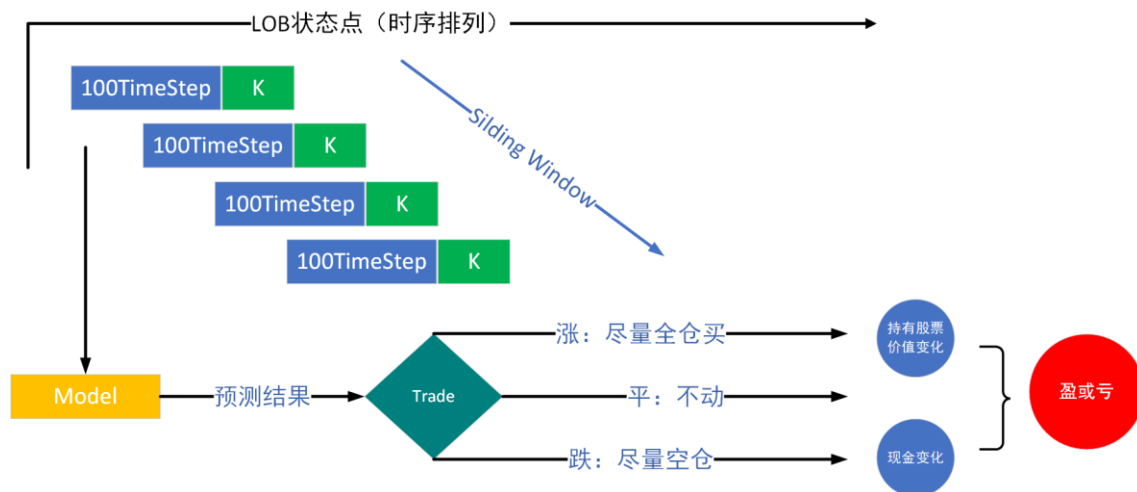


图 5-5 回测实验框架图

在此部分的实验中，分为四组进行，滑动窗口数量分别为 300，500，100，1500，滑动窗口数量越大说明执行策略的时间越长。同样比较了 LC、MLP、CNN、LSTM、初版模型和终版模型的性能。此外，预测范围 K 取 30，展示的结果为在五支股票上实验的平均结果。

5.2.2 实验结果分析

首先展示盈亏曲线图：

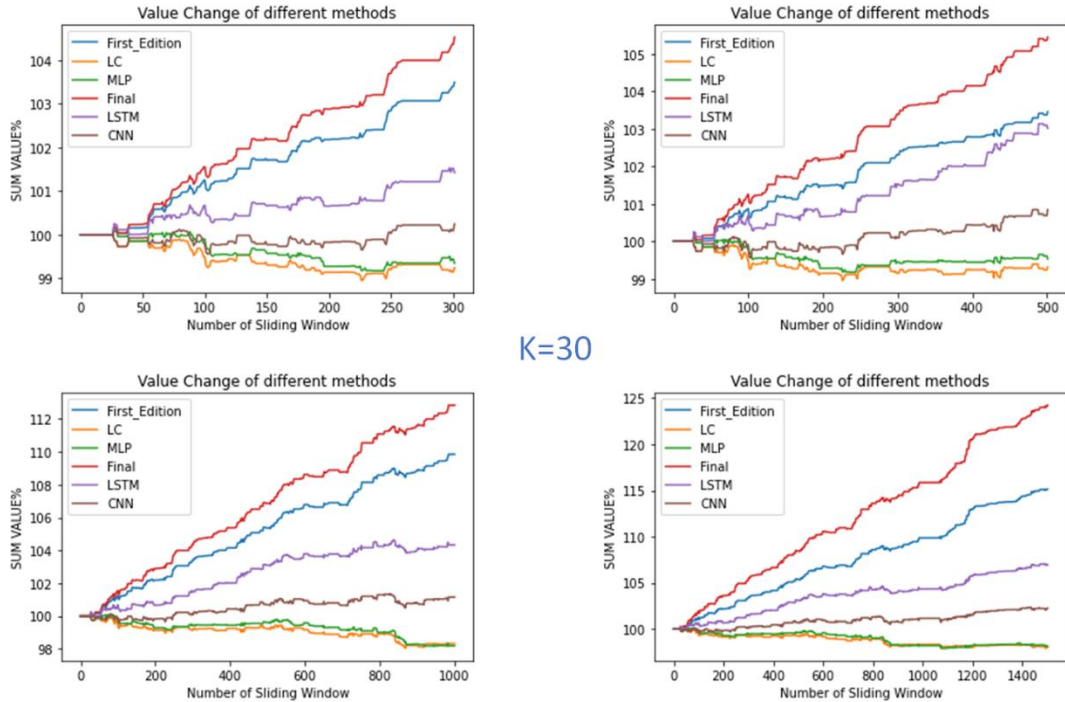


图 5-6 K=30 时回测盈亏结果

回测盈亏结果显示六个模型综合表现评级为：

$$LC \approx MLP < CNN < LSTM < \text{初版模型} < \text{终版模型}$$

这与前一节中预测表现评级保持一致，其中 LC 与 MLP 很难盈利且从总体上看是呈现亏损趋势的；CNN 勉强可以保证不亏损但是盈利能力十分有限，仅有 1% 左右；LSTM 具备一定的盈利能力，当滑动窗口数为 1500 时，即持续模拟交易周期涵盖了 7500 个 x_t 之后，盈利率达到了 5%；初版模型相较于前 4 个模型具备明显更强的盈利能力，当滑动窗口数量为 1500 时，盈利率可以达到 14% 左右；改进后的终版模型则可以接近 25%。

结 论

在本毕业设计中，针对高频交易中的 Limit Order Book MidPrice 涨跌趋势预测这一难点及热点问题进行了研究。经过广泛的领域内已有方法的调研及大量的实验摸索，创新性地设计并实现了初版模型以及改进后的终版模型；为了进一步验证模型的有效性，还设计了基于预测模型的回测框架并加以实现。最后，经过两大部分的充分实验证明了初版模型相较于已有的 LC、MLP、CNN 及 LSTM 分类器具有更好的效果，而终版模型又在初版模型的基础上得到了进一步的提升。将本论文的创新点总结如下：

(1) 对于初版模型的设计：通过文献调研发现前人常常利用 CNN 进行特征提取，同时也有学者尝试过利用 LSTM 抓取股票 LOB 数据之间的时序依赖关系并取得了不错的效果。基于此，设计出了 CNN+LSTM 的基本结构，再从现代深度学习中常用的多通道技巧获得启发，设计出 CNN+三通道+LSTM 的初版模型结构。

(2) 对于终版模型的设计：从模型角度来看，通过引入 seq2seq 架构使得模型可以同时输出多个预测范围 K 下的预测结果，且前一个时间步的预测输出可以作为下一个时间步的输入的一部分。另外使用 cross-attention 机制来生成 Context Vector，这样可以综合考虑我们的 LSTM Encoder 每一个时间步的隐状态输出而不是像初版模型那样仅考虑最后一个时间步的隐状态输出。从特征角度来看，从分析市场买卖双方压力的角度引入两个新的统计特征 VOI 和 OIR。

(3) 对于回测框架的设计：将三种预测结果分别对应到三种买卖动作上，并且设计出了“滑动窗口”风格的回测数据读取方式。考虑到算法运行、网络延迟等时间因素，在买卖动作执行时采用“延迟价格”而不是“即时价格”。

针对这个研究方向，我认为未来可以从以下方向继续探索：

(1) 可以继续阅读文献，查阅一些能够反应 LOB 动态特征的指标，在通过稳

定性验证及与标签相关性检测之后加入到深度学习模型的特征集中，观察是否会得到效果提升。

(2) 本质上来说，做价格预测的最终目的是为了指导交易者优化交易策略从而获取更多的利益，而本毕设的工作重心在于预测模型的设计与优化，交易策略的设计是相对简单的，即尽可能全仓买或者尽力清仓。后续可以引入强化学习框架来寻找每一步买卖动作的最佳数量，即在预测模型与执行交易之间加一个代理 agent 来实现，该 agent 甚至可以学习跨多种股票的投资组合。简言之，朝着这一方向的工作需要强化学习，目前已经有研究工作^[24]证明了 RL 中的 Q 学习在这个领域内的合理性。

参考文献

- [1] E. Zivot and J. Wang, “Vector autoregressive models for multivariate time series,” *Modeling Financial Time Series S-PLUS*, pp. 385–429, 2006.
- [2] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, “Stock price prediction using the ARIMA model,” in *Proc. 16th IEEE Int. Conf. Comput. Model. Simulation 2014*, pp. 106–112.
- [3] G. L. Fol, J. Jasiak, and C. G. Gourieroux, (1999). Intra-day market activity. *Journal of Financial Markets* **2**, 193–226.
- [4] J. P. Bouchaud, M. Mezard, and M. Potters, (2002). Statistical properties of stock order books: Empirical results and models. *Quantitative Finance* **2**, 251–256.
- [5] E. Smith, J. D. Farmer, L. Gillemot, and S. Krishnamurthy, (2003). Statistical theory of the continuous double auction. *Quantitative Finance* **3**, 481–514.
- [6] R. Cont, and A. de Larrard, (2011). Price dynamics in a Markovian limit order market. *SSRN eLibrary*.
- [7] R. Cont, S. Stoikov, and R. Talreja, (2010). A stochastic model for order book dynamics. *Operations Research* **58**, 549–563.
- [8] W. Huang, Y. Nakamori, and S.-Y. Wang, “Forecasting stock market movement direction with support vector machine,” *Comput. Oper. Res.*, vol. 32, no. 10, pp. 2513–2522, Oct. 2005.
- [9] K. J. Kim, “Financial time series forecasting using support vector machines,” *Neurocomputing*, vol. 55, nos. 1–2, pp. 307–319, Sep. 2003.
- [10] L. J. Cao and F. E. H. Tay, “Support vector machine with adaptive parameters in financial time series forecasting,” *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1506–1518, Nov. 2003.
- [11] N. Passalis, A. Tefas, J. Kannianen, M. Gabbouj, and A. Iosifidis, “Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data,” *IEEE Trans. Emerg. Topics Comput. Intell.*, to be published.

- [12] A. Ntakaris, M. Magris, J. Kannianen, M. Gabbouj, and A. Iosifidis, “Benchmark dataset for mid-price prediction of limit order book data with machine learning methods,” *J. Forecasting*, vol. 37, no. 8, 852–866, 2018.
- [13] A. Tsantekidis, N. Passalis, A. Tefas, J. Kannianen, M. Gabbouj, and A. Iosifidis, “Forecasting stock prices from the limit order book using convolutional neural networks,” in *Proc. IEEE 19th Conf. Business Inform.*, 2017, vol. 1, pp. 7–12.
- [14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [15] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, 1994.
- [16] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
- [17] J. Sirignano and R. Cont, “Universal features of price formation in financial markets: perspectives from deep learning,” *arXiv preprint arXiv:1803.06917*.
- [18] C. Carrie, “The new electronic trading regime of dark books, mashups and algorithmic trading,” *Trading*, vol. 2006, no. 1, pp. 14–20, 2006.
- [19] O'Hara, M., & Ye, M. (2011). Is market fragmentation harming market quality? *Journal of Financial Economics*, 100(3), 459–474.
- [20] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.
- [21] A. Lipton, U. Pesavento, and M. G. Sotiropoulos, “Trade arrival dynamics and quote imbalance in a limit order book,” 2013, *arXiv preprint arXiv:1312.0514*.
- [22] Tarun Chordia and Avanidhar Subrahmanyam. Order imbalance and individual stock returns: Theory and evidence. *Journal of Financial Economics*, 72:485–518, 2004.
- [23] Han-Ching Huang, Yong-Chern Su, and Yi-Chun Liu. The performance of imbalance-based trading strategy on tender order announcement day. *Investment Management and Financial Innovations*, 11(2):38–46, 2014.

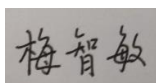
- [24] Sezer, Omer Berat, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu.
"Financial time series forecasting with deep learning: A systematic literature review:
2005–2019." *Applied soft computing* 90 (2020): 106181.

哈尔滨工业大学本科毕业设计（论文）原创性声明

本人郑重声明：在哈尔滨工业大学攻读学士学位期间，所提交的毕业设计（论文）《时间序列大数据压缩与查询》，是本人在导师指导下独立进行研究工作所取得的成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明，其它未注明部分不包含他人已发表或撰写过的研究成果，不存在购买、由他人代写、剽窃和伪造数据等作假行为。

本人愿为此声明承担法律责任。

作者签名：



日期： 2022 年 5 月 24 日

致 谢

2018 到 2022，我在工大的四年来到了尾声。作为一个来自南方的学生，回首这四年的冰城求学旅途，艰辛虽多但收获可贵。正如在完成此毕业设计的过程中，我总共阅读约 50 篇量化金融与机器学习期刊及会议学术论文，同时入门 Pytorch 深度学习框架并对 10 余种模型框架有了更深入的理解。更重要地，与此前多次与同学组队参加各种竞赛不同，本毕设是我第一次独立地、完整地完一次学术研究，这让我收获颇多。

在本次毕业设计过程中，许多人给予过我帮助，我在此依次感谢：

首先要感谢的是我的毕设指导老师张彦航和刘旭东，他们每隔两周左右就会给我们开一次例会从而督促我们的毕设进度；同时他们提醒我不仅要考虑模型的预测表现，还要补充回测实验来进一步说明模型的盈利能力；此外，当我遇到 PC 算力不够的问题时，他们给我免费提供了实验室的服务器以帮助训练模型；在开题、中期和结题之际，他们也会针对我提交的报告初稿提出细致的修改意见。

其次需要感谢我的很多同学，尤其是几位仍有联系的高中同学和大一就认识的几位朋友，在此就不逐一列出姓名了。十分感谢他们的陪伴成长，并且在我毕设进度缓慢之时和我电话交流，这大大缓解了我的压力。所谓“青春最美的不是梦，而是与你一起追梦的人”，我一直相信并会继续相信这句话。

最后，我需要郑重感谢我的父母，他们虽然没上过学，知识水平不高，但总是最为关心我的学习情况和心理健康，并且每周和我通电话询问我在学校的状态，在我最繁忙的 5 月份常常给予我劝导，让我耐心做完毕设和港校的课题并且让我不要担心家里的事情。

行文至此，感慨颇多，但也无需多言。愿每个人都可以找到属于自己的闪耀并努力追逐，我谓之不负自己！