K 线图时序分析-价格预测的探索与应用

ZWN

摘要

金融时间序列预测，无论是分类还是回归，都是过去十年的热门研究课题。虽然传统的机器学习算法效果平平，但深度学习在很大程度上促进了预测性能的提升。由于股价数据具有高频、非线性、长记忆的特点，准确预测股价是一项挑战。目前已经提出了各种预测方法，从经典的时间序列方法到基于机器学习的方法，例如随机森林（RF）、循环神经网络（RNN）、卷积神经网络（CNN）、长短期记忆（LSTM）神经网络及其变体等。每种方法都可以达到一定的准确度，但也有其局限性。本文提出了一种基于 CNN-BiLSTM-Attention 的模型来提高预测股价和指数的准确性。首先，利用卷积神经网络（CNN）和双向长短期记忆（BiLSTM）网络提取序列数据的时间特征。然后，引入注意机制来自动对信息特征进行权重分配；最后，通过密集层输出最终的预测结果。

本文可视作对该研究 [1] 的复现，该研究中并未给出其模型的实现，本工作尝试探索了此种模型架构的实现，证明了其在预测上的准确度，并进行了应用上的探索与模型上的优化，最终实现一种轻量级、易部署、高准确度的模型。代码可在 Github 查看: https://github.com/ZWN2001/CNN-BiLSTM-Attention-K-Line-Prediction

1 研究概况

股票市场是金融市场的重要组成部分，股票市场的大幅波动会对经济产生较大的不利影响 [2]。如果能够较为准确地预测股票价格，可以通过有针对性的措施避免股市崩盘，引导股票市场良好运行，最终为金融市场的健康发展奠定更坚实的基础 [3]。因此，股票市场内在价值及预测的研究越来越受到学者和实践者的重视，并取得了一系列成果 [4]。

股票市场波动剧烈、无规律可循，其股票数据呈现出数据量大、信息模糊、非线性、非平滑等复杂特点。传统计量经济学方法如自回归移动平均法（ARMA）、广义自回归条件异方差法（GARCH）等对波动性较小的数据有较好的预测效果 [5–7]。传统的机器学习算法如随机森林、支持向量机等可以很好地学习股票与各类影响因素之间的非线性关系。但这些方法在模型构建过程中过于依赖样本的选择，导致模型构建和更新不够灵活，预测精度难以满足预测要求 [8, 9]。深度学习模型比传统机器学习模型具有更强的学习能力和自适应能力，在股票价格分析中表现出色 [10]。

长短期记忆（LSTM）神经网络是一种循环神经网络，它可以更好地处理长输入序列，同时考虑到股票数据的时间序列和非线性特性。因此，与只能记住短序列的其他循环神经网络相比，LSTM 凭借出色的内存容量和门结构，广泛应用于股票预测 [11]。Moghar 和 Hamiche 提出了一种基于 LSTM 的循环神经网络（RNN）模型来预测 GOOGL 和 NKE 的开盘价趋势，测试结果验证了该模型的有效性 [12]。Vidal 和 Kristjanpoller 提出了一种混合 CNN 和 LSTM 来预测黄金价格波动，并得出结论，该模型可以充分提取时间序列特征以进行高精度预测。预测结果优于单独的 CNN 和 LSTM 模型 [13]。Nelson 等人使用 LSTM 网络根据历史价格和技术分析指标预测股价未来趋势。实验结果表明，该方法平均准确率达到 55.9% [14]。

BiLSTM 是一种双向长短期记忆网络，它是传统单向 LSTM 的扩展，利用其学习双向

时间序列特征的能力进一步提高模型预测精度。Jia 等人使用 BiLSTM 模型预测 GREE 股票价格，结果表明 BiLSTM 比 LSTM 模型提高了预测精度 [15]。Wang 等人提出了一种组合 CNN-BiSLSTM 模型（对 BiLSTM 模型的改进）用于股票价格预测，并将其与 LSTM 模型、BiLSTM 模型、CNN-LSTM 模型和 CNN-BiLSTM 模型进行了比较，得出结论，所提出的模型是这些模型中最好的 [16]。

神经网络中的注意力机制是一种资源分配方案，用于在计算能力有限的情况下将计算资源分配给更关键的任务，同时解决信息过载问题。Cinar 等人提出了一种针对 RNN 的扩展注意力模型。实验结果表明，带有 RNN 的模型可以捕捉时间序列中的伪周期，其扩展性能明显优于原模型 [17]。Wang 等人提出了一种基于二次分解（SD）、多因素分析（MFA）和基于注意力的长短期记忆（ALSTM）的混合模型，用于预测亚洲四个主要国家的股市价格趋势。实证分析结果表明，与一般的长短期记忆相比，所提出的模型可以获得至少高 30% 的准确率，证明了混合模型的有效性 [18]。

2 问题建模

在一段连续的时间内，某只股票会产生一个收盘价格的序列。我们记这段连续的价格序列为：

$$C_T = c_1, c_2, ..., c_T$$

其中 c_t 代表该股票在第 t 个交易日的收盘价格。

此外我们定义开盘价序列 $O_T = o_1, o_2, ..., o_T$，最高价序列 $H_T = h_1, h_2, ..., h_T$，最低价序列 $L_T = l_1, l_2, ..., l_T$

基于这四个序列数据，训练一个模型 f_M，以第 $T+1$ 天的开盘价 o_{T+1}、最高点 h_{T+1}、最低点 l_{T+1} 作为输入，预测该天的收盘价格 c_{T+1}

即：

$$\{O_T, H_T, L_T, C_T\} \rightarrow f_M$$

$$c_{T+1} = f_M(o_{T+1}, h_{T+1}, l_{T+1})$$

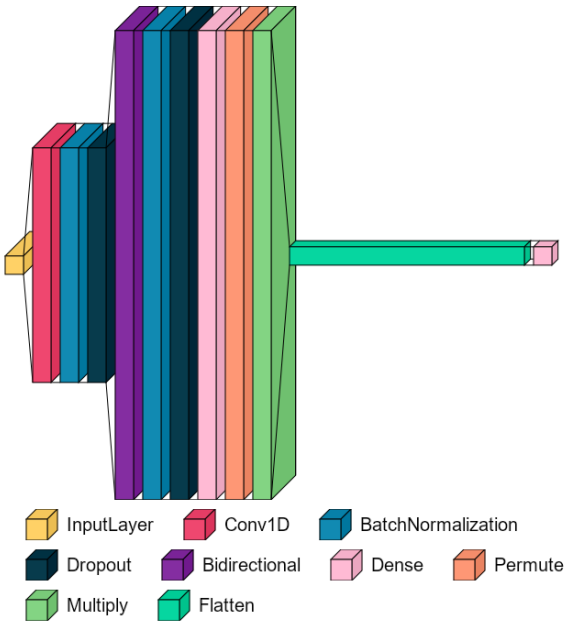

图 1: 基础模型结构的可视化表示.

考虑更现实一点的，训练一个模型 f_N，以前 t 天数据为输入，预测第二天的开盘价 o_{T+1}、最高点 h_{T+1}、最低点 l_{T+1} 、收盘价 c_{T+1}

$$\{(O_i, H_i, L_i, C_i)\} \rightarrow f_N, i \in (0, T-1)$$

$$(c_{T+1}, o_{T+1}, h_{T+1}, l_{T+1}) = f_N(c_T, o_T, h_T, l_T)$$

本工作主要聚焦于第二类问题，即通过第 T 天的数据，预测第 $T+1$ 天的 K 线图数据。

3 模型结构

3.1 结构概述

如图1所示，首先利用 CNN 层提取股票数据内部特征，CNN 层由 1D 卷积层、归一化层和 Dropout 层组成。然后，在 CNN 提取的局部特征上训练 BiLSTM 层 (及其相应的归一化层，Dropout 层，Dense 层)，学习内部动态变化模式。在此基础上，引入注意力机制，包括 Permute 层、Multiply 层、Flatten 层，自动为 BiLSTM 层提取的特征赋予不同的权重，探索深度时间相关性。最后经过一个 Dense 层得到输出结果。模型具体结构及其可视化结构图可参见附录。

3.2 相对于原文的模型结构与体积优化

原文中使用的模型结构为Conv2D+BiLSTM+Attention+Dense 的模型结构，如图2所示，该研究结果中存在大量的预测准确度问题，通常表现在价格波动中普遍的的高峰与低谷价格预测不准确，本工作的结构优化主要针对这一问题进行展开。

本工作对该模型进行了如下优化：

- 使用 Conv1D 替代原研究中的 Conv2D
- 增加归一化与更多 DropOut
- 使用蒸馏、量化对模型进行优化

首先，本工作使用 Conv1D 替代原研究中的 Conv2D，这是因为时序数据是线性的，适合一维卷积的滑动窗口机制，而不是二维卷积所需的空间关系。一维卷积直接针对时间序列的结构进行建模，计算效率高，且能够有效捕捉时间依赖性，而二维卷积在时间序列中可能过于复杂且不直观，容易导致过拟合。虽然在研究股价与其他相关变量（如成交量、技术指标）之间复杂关系时，二维卷积可能会有一定作用，但这不是主流方法。（原研究中并没有提及二维数据的构造方法，所以也难以探究原研究中的二维数据是如何得来的，且原研究中并未公开数据集）

其次，本工作在原有工作基础上增加归一化与更多 DropOut 层，这一变动的主要目的是改善模型的过拟合倾向。

一方面，本工作对输入数据进行归一化操作，在输出时进行反归一化。一般来说，这一操作的主要目的是让所有特征值处于同一个量级，从而确保每个特征对模型的贡献是均衡的。不过本工作使用到的特征值在量级上并没有明显差别，所以这一操作的主要目的还是让权重更新更加平稳，从而加速模型的训练过程。以及，如果拓展模型输入特征空间（如引入交易量等其他量级的特征时），更方便模型的过渡。

另一方面，本工作引入了更多的归一化层。在深度神经网络中，梯度可能会因为网络层数过多而逐层减小（梯度消失）或者逐层增大（梯度爆炸），归一化层通过调整激活值的分布，确保它们处于一个合理的范围，能够有效缓解梯度问题。最后，更多 Dropout 层的引入可以在一定程度上改善模型过拟合的问题，这一问题实际上是非常容易发生的。

最后，本工作针对模型进行了蒸馏与量化，其主要目的是在不损失精度的情况下，大幅压缩模型体积、降低模型复杂度，同时也可视为一种正则化，改善模型过拟合问题，最终达到一种易于部署的形态。

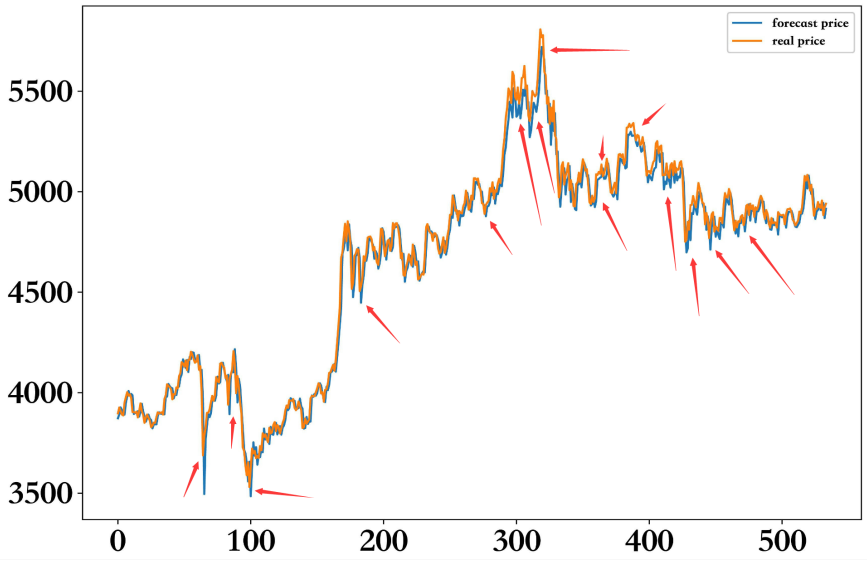

图 2: 原研究中存在的拟合效果问题.

4 实验

由于原研究并未公布数据集及 Conv2D 数据的构造方法，所以后续主要介绍本工作所涉及模型的实验过程及效果评估。

4.1 超参数

本工作中涉及的部分关键超参数如下：

- 时间步长：20
- 批次大小：64
- LSTM 每个方向的隐藏单元数量：64
- 一维卷积卷积核（滤波器）数量：64
- Dropout 层中随机丢弃神经元的比例：0.4
- 训练轮数：30
- 学习率：0.001
- 学习率衰减步长：1000
- 学习率衰减率：0.96

4.2 数据预处理

4.2.1 数据归一化与反归一化

遍历每列数据，计算最小值和最大值，根据公式将数据归一化到 [0, 1]，同时记录每列的 [min, max]。公式如下：

对于输入矩阵 data，设第 i 列的数据为 $x_1, x_2, ..., x_n$，其最小值为 min_i，最大值为 max_i。第 j 行第 i 列的数据归一化后的值为：

$$x'_{j,i} = \frac{x_{j,i} - \min_i}{\max_i - \min_i}, \quad \max_i \neq \min_i$$

反归一化操作中，遍历每列归一化数据，根据公式使用记录的 [min, max] 将其还原到原始范围。公式如下：

对于归一化后的数据矩阵 data_norm，设第 i 列的原始最小值为 min_i，最大值为 max_i。第 j 行第 i 列的数据反归一化后的值为：

$$x_{j,i} = x'_{j,i} \cdot (\max_i - \min_i) + \min_i, \max_i \neq \min_i$$

4.2.2 LSTM 数据构造

如算法 1 所示，该算法使用滑动窗口生成输入序列和对应的目标值。具体来说，look_back 参数定义了每个输入序列的时间步数（即窗口大小）。代码通过遍历时间序列 dataset，每次从数据中提取长度为 look_back 的连续切片作为输入特征 dataX，而窗口后一个时间步的数据（i + look_back 的数据点）作为目标值 dataY。最终，输入特征和目标值分别被转换成 NumPy 数组 Train_X 和 Trai_Y，以便 LSTM 模型能够使用这些有监督的时间序列数据进行训练。这种方法可以帮助 LSTM 学习基于过去 look_back 个时间步预测下一个时间步的数据。

Algorithm 1 Create Dataset for LSTM Training

Require: *dataset* (time-series data with shape (n, m)), *look_back* (window size)
Ensure: $TrainX$, $TrainY$ (input sequences and corresponding targets)
$dataX \leftarrow []$
$dataY \leftarrow []$
for $i \leftarrow 0$ to $length(dataset) - look\_back - 1$ **do**
  $a \leftarrow dataset[i : i + look\_back, :]$
  Extract input sequence of length $look\_back$
  $b \leftarrow dataset[i + look\_back, :]$
  Extract target value (next time step)
  $dataX.append(a)$
  $dataY.append(b)$
end for
$TrainX \leftarrow$ convert $dataX$ to array
$TrainY \leftarrow$ convert $dataY$ to array
return $TrainX, TrainY$

4.3 实验过程

首先，本工作基于图1所示的基础模型进行模型评估。在此基础上，对该模型进行蒸馏、量化。其中量化使用现有的库函数实现，这里着

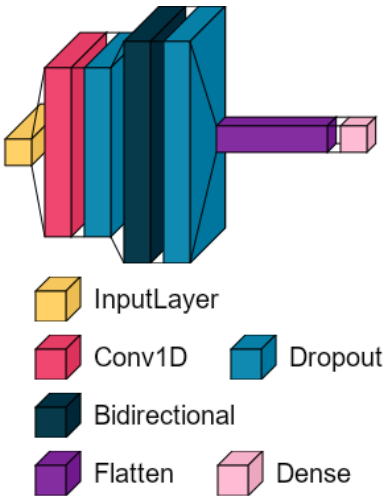

图 3: 学生模型结构的可视化表示.

重介绍一下蒸馏的实现细节。

蒸馏的核心思想是通过将教师模型的输出信息（通常是软标签，soft labels）传递给学生模型。软标签不同于硬标签，它包含了更多关于类别之间关系的信息。教师模型的输出通常会比学生模型的输出更平滑、更有信息，因此能够帮助学生模型学到更为细致的特征表示。

本工作中蒸馏采用基础模型作为教师模型，采用图3所示的模型结构作为学生模型。相较于教师模型，学生模型使用的一维卷积卷积核（滤波器）数量为教师模型的二分之一，使用的 LSTM 每个方向的隐藏单元数量为教师模型的三分之一，没有归一化层与 Attention 层，参数量约为教师模型的七分之一。模型具体结构及其可视化结构图可参见附录。

使用到的蒸馏损失函数如下

$$L_{distill} = \alpha \cdot D_{KL}(q(T), p(T)) + (1-\alpha) \cdot L_{hard}(y, \hat{y})$$

其中 $D_{KL}(q(T), p(T))$ 是教师和学生模型的输出之间的 KL 散度，$L_{hard}(y, \hat{y})$ 是基于硬标签的损失（通常是交叉熵），α 是调节硬标签损失与蒸馏损失之间权重的超参数。

	MAE	MSE	涨跌准确率 (%)
基础模型	10.91	15.8	98.8192
蒸馏模型	2.38	1.32	98.4383
模型量化	3.21	1.82	98.4574

表 1: 本工作涉及模型的评估结果，MAE 量级为 10^{-3}，MSE 量级为 10^{-5}

	基础模型	蒸馏模型	蒸馏模型量化
大小	396844	67668	67540

表 2: 模型 TFLite 文件大小，单位为字节

5 实验结果评估

5.1 评价指标

本工作选用 MAE、MSE 以及价格涨跌的准确率作为评价指标，其中 MAE 与 MSE 计算公式如下：

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

5.2 实验结果

实验结果如表1所示，基础模型在测试集的表现相对较差，蒸馏后模型在准确度上得到了较大提升，但随着模型体积的优化，模型效果相对降低，但都在可接受的范围内。

模型体积优化结果如表2所示，通过蒸馏、量化操作，在模型精度损失可接受的条件下，实现了对基础模型的大幅压缩，压缩比约为 5.87，且最终模型大小已经基本符合在移动端等低算力平台部署的要求。（由于量化是使用现有的库函数实现的，所以在效果上表现还需要进行改善）

5.3 结果可视化分析

针对测试集拟合结果，该部分内容主要对其结果可视化进行分析，并讨论其问题与出现

的原因。图4、5、6依次展示了不同阶段模型在测试集的部分拟合结果。可以看到，基础模型与原有研究的模型类似，在峰值与低谷的价格拟合上表现得都不是很好，这说明模型还是存在一定的过拟合倾向。这是因为在股票交易记录中，大部分数据都集中在平均值附近，而峰值或低谷数据较少，所以过拟合会导致论文更偏向与“保守化”的数据均值，导致在峰值与低谷的价格拟合上表现不佳。

蒸馏（图5）、量化（图6）后的模型拟合效果都非常不错，但在峰值与低谷的价格上仍然有一定的拟合不佳的问题，但相对于基础模型有较大的改善。

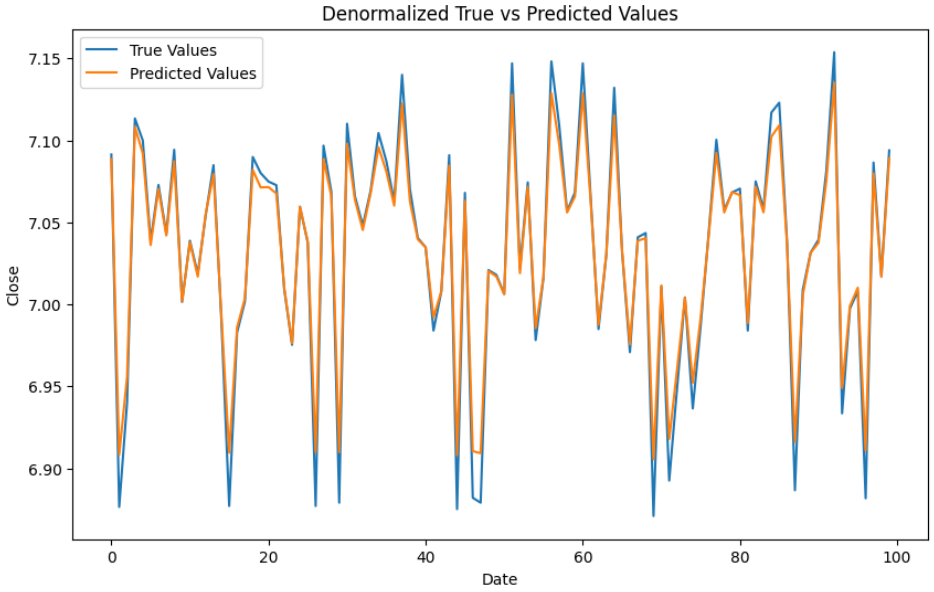

图 4: 基础模型拟合结果.

9

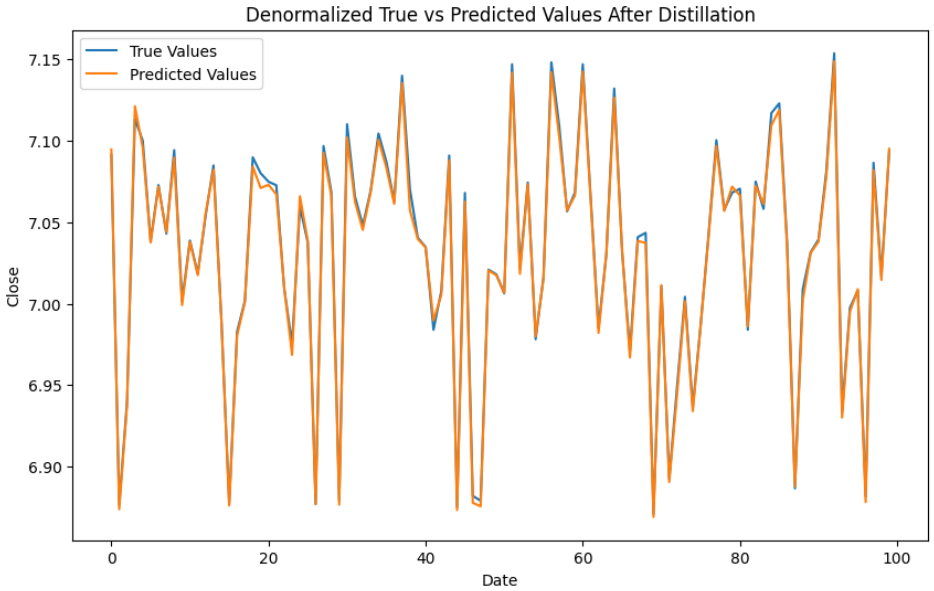

图 5: 蒸馏模型拟合结果.

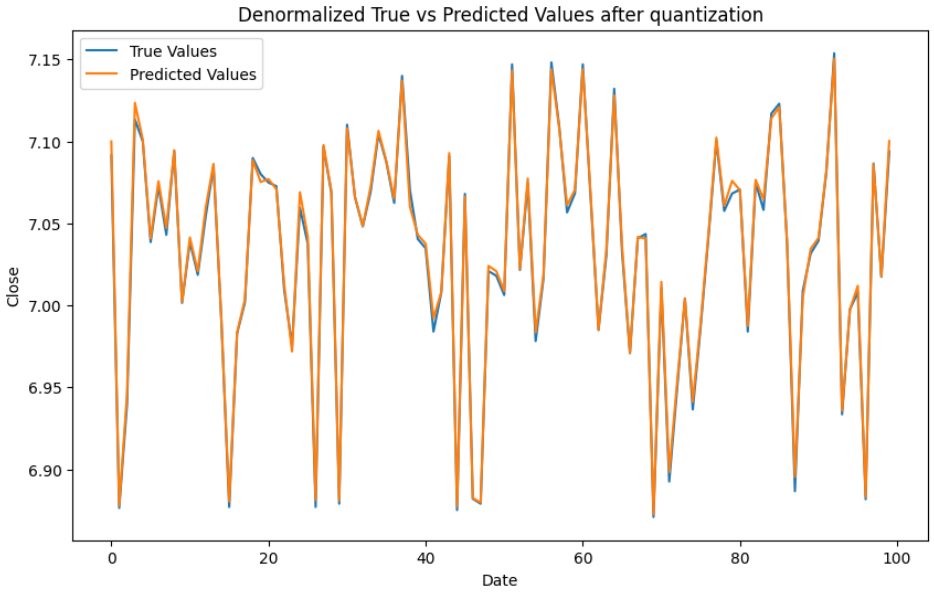

图 6: 量化后拟合结果.

9

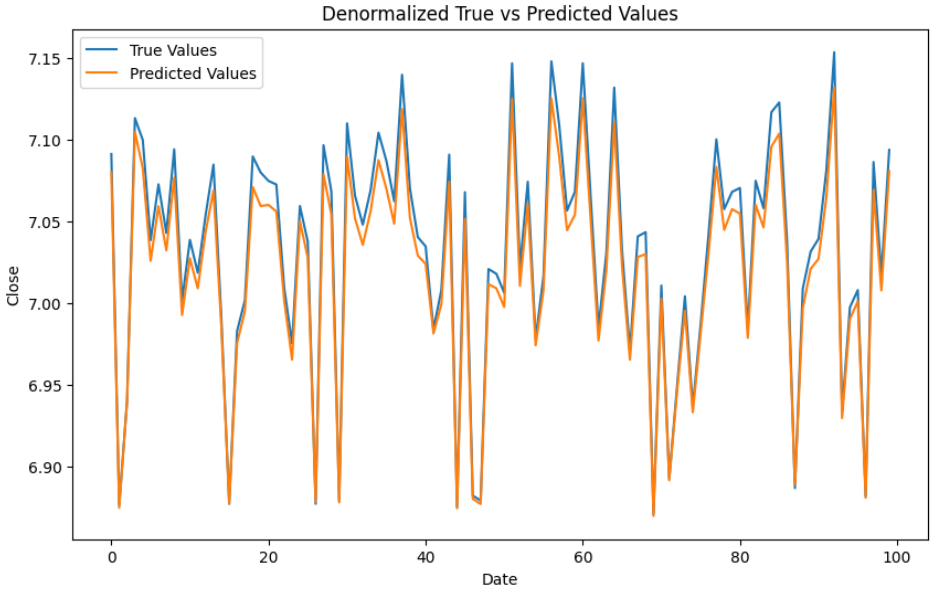

图 7: 直接训练学生模型.

7 结论

本工作主要对原有研究模型进行了改进，同时在此基础上进行了蒸馏与量化，最终优化模型体积到一个较小的数值，且精度上的损失在可接受范围内，实现了较好的效果。

同时，本工作也表明，原研究在模型结构上存在缺陷，尤其是过于复杂的数据处理及其二维卷积可能是造成其过拟合的问题所在。此外，本工作进行的蒸馏也实现了在提升精度的同时大幅压缩了模型体积，取得了非常不错的优化效果。

对于蒸馏为什么有效。我认为是基础模型

	MAE($\times 10^{-3}$)	MSE($\times 10^{-5}$)	涨跌准确率 (%)
基础模型	10.91	15.8	98.8192
蒸馏模型量化结果	3.21	1.82	98.4574
直接训练学生模型	15.17	29.86	98.4574

表 3: 本工作涉及模型的评估结果

的注意力机制对整体模型参数的训练进行了优化，通过蒸馏可以将这一整体性的优化结果压缩到一个没有注意力的模型中，从而在保留原有的模型效果的基础上实现模型的简化。

8 不足与改进思路

首先，本工作实际上进行了剪枝，但并没有取得很好的结果，究其原因还是没有能够很好的使用现有函数库。

其次，量化操作实际上也可以进行进一步优化，比如进行更激进的 INT8 量化，可以进一步的压缩模型体积。

再者，可以针对不同的超参数进行更详细的自动超参数优化方案（比如使用贝叶斯），从而使模型具有更好的健壮性，能够实现更好的泛用性。

最后，该工作仅使用了股价的价格数据，可以考虑使用更广泛的特征空间，如将交易量、舆情等纳入考量，会有更大的数据挖掘空间。

参考文献

[1] Jilin Zhang, Lishi Ye, and Yongzeng Lai. Stock price prediction using cnn-bilstm-attention model. *MATHEMATICS*, 11(9), APR 23 2023.

[2] Yu Wei and Peng Wang. Forecasting volatility of ssec in chinese stock market using multifractal analysis. *Physica A: Statistical Mechanics and its Applications*, 387(7):1585–1592, 2008.

[3] Pingan Wang, Yuanwei Lou, and Lei Lei. Research on stock price prediction based on bp wavelet neural network with mexico hat wavelet basis. In *2017 International Conference on Education, Economics and Management Research (ICEEMR 2017)*, pages 99–102. Atlantis Press, 2017.

[4] Yaping Hao and Qiang Gao. Predicting the trend of stock market index using the hybrid neural network based on multiple time scale feature learning. *Applied Sciences*, 10(11):3961, 2020.

[5] Subhash Arun Dwivedi, Amit Attry, Darshan Parekh, and Kanika Singla. Analysis and forecasting of time-series data using s-arima, cnn and lstm. In *2021 international conference on computing, communication, and intelligent systems (icccis)*, pages 131–136. IEEE, 2021.

[6] Sarbjit Singh, Kulwinder Singh Parmar, and Jatinder Kumar. Soft computing model coupled with statistical models to estimate future of stock market. *Neural Computing and Applications*, 33(13):7629–7647, 2021.

[7] Ying Xiang. Using arima-garch model to analyze fluctuation law of international oil price. *Mathematical Problems in Engineering*, 2022(1):3936414, 2022.

[8] Huseyin Ince and Theodore B Trafalis. Kernel principal component analysis and support vector machines for stock price prediction. In *2004 IEEE International Joint Conference on Neural Networks*

(IEEE Cat. No. 04CH37541), volume 3, pages 2053–2058. IEEE, 2004.

[9] Lili Yin, Benling Li, Peng Li, and Rubo Zhang. Research on stock trend prediction method based on optimized random forest. *CAAI Transactions on Intelligence Technology*, 8(1):274–284, 2023.

[10] Lin Sun, Wenzheng Xu, and Jimin Liu. Two-channel attention mechanism fusion model of stock price prediction based on cnn-lstm. *Transactions on Asian and Low-Resource Language Information Processing*, 20(5):1–12, 2021.

[11] S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[12] Adil Moghar and Mhamed Hamiche. Stock market prediction using lstm recurrent neural network. *Procedia computer science*, 170:1168–1173, 2020.

[13] Andrés Vidal and Werner Kristjanpoller. Gold volatility prediction using a cnn-lstm approach. *Expert Systems with Applications*, 157:113481, 2020.

[14] David MQ Nelson, Adriano CM Pereira, and Renato A De Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. Ieee, 2017.

[15] Mingzhu Jia, Jian Huang, Lihua Pang, and Qian Zhao. Analysis and research on stock price of lstm and bidirectional lstm neural network. In *3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2019)*, pages 467–473. Atlantis Press, 2019.

[16] Haiyao Wang, Jianxuan Wang, Lihui Cao, Yifan Li, Qiuhong Sun, and Jingyang Wang. A stock closing price prediction model based on cnn-bislstm. *Complexity*, 2021(1):5360828, 2021.

[17] Yagmur Gizem Cinar, Hamid Mirisaee, Parantapa Goswami, Eric Gaussier, Ali Aït-Bachir, and Vadim Strijov. Position-based content attention for time series forecasting with sequence-to-sequence rnns. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, Part V 24*, pages 533–544. Springer, 2017.

[18] Jujie Wang, Quan Cui, Xin Sun, and Maolin He. Asian stock markets closing index forecast based on secondary decomposition, multi-factor analysis and attention-based lstm model. *Engineering Applications of Artificial Intelligence*, 113:104908, 2022.

9 附录

模型结构可视化图过大，可参见 Github 仓库。模型细节如下：

```
Model: "model_8"
_________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================================================
 input_9 (InputLayer)        [(None, 20, 4)]           0         []

 conv1d_8 (Conv1D)           (None, 20, 64)            320       ['input_9[0][0]']

 batch_normalization_8 (Bat  (None, 20, 64)            256       ['conv1d_8[0][0]']
 chNormalization)

 dropout_16 (Dropout)        (None, 20, 64)            0         ['batch_normalization_8[0][0]'
                                                                 ]

 bidirectional_5 (Bidirecti  (None, 20, 128)           66048     ['dropout_16[0][0]']
 onal)

 batch_normalization_9 (Bat  (None, 20, 128)           512       ['bidirectional_5[0][0]']
 chNormalization)

 dropout_17 (Dropout)        (None, 20, 128)           0         ['batch_normalization_9[0][0]'
                                                                 ]

 dense_12 (Dense)            (None, 20, 128)           16512     ['dropout_17[0][0]']

 attention_vec (Permute)     (None, 20, 128)           0         ['dense_12[0][0]']

 multiply_4 (Multiply)       (None, 20, 128)           0         ['dropout_17[0][0]',
                                                                  'attention_vec[0][0]']

 flatten_8 (Flatten)         (None, 2560)              0         ['multiply_4[0][0]']

 dense_13 (Dense)            (None, 4)                 10244     ['flatten_8[0][0]']

=================================================================================================
Total params: 93892 (366.77 KB)
Trainable params: 93508 (365.27 KB)
Non-trainable params: 384 (1.50 KB)
_________________________________________________________________________________________________
```

图 8: 基础模型结构.

```
Model: "model_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_10 (InputLayer)       [(None, 20, 4)]           0

 conv1d_9 (Conv1D)           (None, 20, 32)            160

 dropout_18 (Dropout)        (None, 20, 32)            0

 bidirectional_6 (Bidirecti  (None, 20, 42)            9072
 onal)

 dropout_19 (Dropout)        (None, 20, 42)            0

 flatten_9 (Flatten)         (None, 840)               0

 dense_14 (Dense)            (None, 4)                 3364

=================================================================
Total params: 12596 (49.20 KB)
Trainable params: 12596 (49.20 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

图 9: 学生模型结构.