南京大学

本 科 毕 业 设 计

院　　系 匡亚明学院

专　　业 统计学

题　　目 基于 LSTM 的时间序列预测

年　　级 2016 级　学 号 161240083

学生姓名 叶梦婕

指导教师 范红军　职 称 副教授

提交日期 2020 年 6 月 17 日

南京大学本科生毕业论文（设计、作品）中文摘要

题目：基于LSTM的时间序列预测

院系：匡亚明学院

专业：统计学

本科生姓名：叶梦婕

指导教师（姓名、职称）：范红军副教授

摘要：

时间序列是指一种按照时间先后顺序排列的数据点，在各行各业应用广泛，金融时间序列，如期货、股票、债券、汇率等等，更是与我们的生活息息相关，而股票市场对于国家的经济发展有着至关重要的作用。本文主要研究对于美国股票市场十家公司的日收盘价这一时间序列的预测。

传统的统计学方法，如ARIMA等，在预测时间序列时，往往只能考虑到时间这个单一的维度，从而造成信息利用率低。近年来，随着计算机技术的发展，基于机器学习与深度学习的研究方法开始兴起。本文利用深度学习中循环神经网络的变体长短期记忆（Long Short Term Memory，LSTM）进行时间序列的预测。

在模型训练前的特征工程过程中，通过为原有数据添加不同的特征组合以及增加训练集样本数的方式，使得最终训练出的模型具有更强的泛化能力。本文将不同特征组合以及不同训练集样本数对于模型最终的预测效果进行对比，说明特征工程的重要性。

实验结果表明，充足的训练数据量可以保证LSTM模型具有较优的预测效果，说明深度学习的方法对于时间序列预测具有启发性意义。

关键词：时间序列预测；特征工程；LSTM；深度学习

南京大学本科生毕业论文（设计、作品）英文摘要

THESIS：Time Series Forecast with Long Short Term Memory

DEPARTMENT： Kuang Yaming Honors School

SPECIALIZATION: Statistics

UNDERGRADUATE: Mengjie Ye

MENTOR: Prof. Hongjun Fan

ABSTRACT:

Time series refers to a sort of data points arranged in chronological order. It is widely used in various industries. Financial time series, such as futures, stocks, bonds, exchange rates, etc., are closely related to our lives. Stock market plays a vital role in the economic development. This paper mainly studies the time series forecast of daily closing prices of ten companies in the US stock market.

Traditional statistical methods, such as ARIMA, often only consider the single dimension, time t, when predicting time series, resulting in low information utilization. In recent years, with the development of computer technology, research methods based on Machine Learning and Deep Learning have begun to emerge. This paper uses the Long Short Term Memory (LSTM), a variant of the Recurrent Neural Network in Deep Learning to predict time series.

In the feature engineering process before model training, by adding different features to the original data and increasing the number of samples in the training set, the trained model could have a stronger generalization ability. This article compares the influences of different features and different sizes of training set on the final prediction to illustrate the significance of feature engineering.

The experimental results show that the sufficient amount of training data can ensure that the LSTM model has a good generalization ability, indicating that the deep learning method is instructive for time series forecast.

KEY WORDS: Time Series Forecast, Feature Engineering, LSTM, Deep Learning

目录

绪论

(一)研究背景及意义

通过一系列时间点上的观测来获取数据是很常见的行为，而时间序列（Time Series）则是指一组按照时间先后顺序排列形成的数据点序列。时间序列数据在各行各业有着广泛的应用：商业上对于周利率、月价格指数、日股票闭盘价、年销售量等的观测；气象上对于每天的最高和最低气温和年降水量与干旱指数等的观测；农业上对于每年作物和牲畜产量、出口销售等数字的记录；生物科学上对每毫秒的心电活动的观察；生态上对动物种群数量变化的记录。由此可以看出对于时间序列数据研究的重要性。通过对于该类数据的分析与预测，我们可以发现其中的潜在规律，从而为决策者提供前瞻性的意见和具有指导性的策略，以应对未来可能发生的状况。

金融时间序列包括期货、利率、债券、股票等等。股市对于国家的经济发展有着至关重要的作用，它关系到整个国家经济的命脉，对于调控宏观经济、优化资源配置等方面起到了不可或缺的作用。因此预测股票市场的趋势十分重要，同时也具有很大的挑战性。因为金融市场是一个复杂而又庞大的系统，受政治、文化、社会等各种因素的影响，它瞬息万变，并伴随着高风险高收益，而金融时间序列又有着非线性、非平稳性、序列相关性等特点。面对变幻莫测的股票市场，投资者需要尽可能在合适的时机进行股票交易，而对于大多数人而言，掌握合适的入市出市时机并不是一件容易的事情，所以对于股市趋势的准确预测对投资者来说意义重大。

传统的分析时间序列的方法是利用统计学的知识对其发展趋势进行建模，利用得到的模型对研究对象进行合理的延伸，从而达到预测的效果。然而，这种方法并没有考虑到外界因素与研究对象之间的因果关系，只是从时间这一维度来建模。因此，当某些影响因素发生重大变化时，传统方法的预测精度将会大幅度降低。现代时间序列预测可以利用机器学习与深度学习的技术进行建模。通过将时间序列的影响因素特征化，分析特征与预测对象之间的因果关系，克服了传统方法的弊端，从而可以大大提高预测的精度。深层神经网络它能够以任意期望的精度逼近连续函数，对于非线性模型有着良好的研究能力。其中，循环神经网络（Recurrent Neural Networks，RNN）在股票价格预测中有良好的表现，作为 RNN

的变体，长短时记忆（Long-Short-Term Memory，LSTM）也常用来解决金融时间序列的预测问题。

(二)国内外文献综述

《时间序列分析：预报与控制》作为 George E.P.Box 和 Gwilym M.Jenkins 的奠基性著作，自它的问世以来，时间序列分析的理论研究和实践都有了飞速的发展。作为数据挖掘领域的重要的研究方向之一，时间序列分析大致可以分为四个方面：时间序列分类、时间序列聚类、时间序列预测与时间序列异常检测。本文主要研究时间序列的预测问题。

对于时间序列预测问题，目前主要的方法有以下三大类：传统的方式是基于统计学的相关理论，而近年来，基于机器学习和神经网络的方法有了很大发展。

早期的时间序列预测问题，主要利用统计学的方法。由于此方法具有扎实的统计学理论基础，得到的模型可靠稳定。1927 年，英国统计学家 George Udny Yule 提出了自回归模型（Autoregressive Model，AR 模型），标志着时间序列预测研究方法的开始。1931 年 Gilbert Walker 提出移动平均模型（Moving Average Model，MA 模型），并在之后将 AR 模型与 MA 模型整合为自回归移动平均模型（Autoregressive Moving Average Model，ARMA 模型），这三者构成了传统时间序列预测的基础，用于处理单变量、同方差、线性的平稳序列数据。1970 年，差分整合移动平均自回归模型（Autoregressive Integrated Moving Average Model，ARIMA 模型）被提出，通过差分的方法来处理非平稳序列。对于多变量、异方差和非线性的情况，有 1976 年提出的动态回归模型（ARIMAX）、1982 年提出的自回归条件异方差（Autoregressive Conditional Heteroskedasticity，ARCH）及其推广 GARCH，和 1978 年提出的门限自回归模型（Threshold Autoregressive Model，TAR 模型）。

对于非线性的时间序列预测，以上统计学模型表现并不理想。上世纪 90 年代以来，随着机器学习技术的兴起与发展，研究人员将重点放在数据间的量化关系上，通过机器学习算法来解决此类问题。常用的包括：支持向量机（Support Vector Machine，SVM）、贝叶斯网络（Bayesian Network）、随机森林（Random Forest）和自适应增强（Adaptive Boosting，Adaboost）等算法。然而，这些方法

会忽略很多时间序列内在的如数据相关性等本质问题。

近年来，深度学习的快速发展，人们开始关注人工神经网络（Artificial Neural Network，ANN）的应用。传统机器学习方法的效果好坏主要取决于特征的选取，而神经网络可以自主学习特征，不需人为选择。同时，由于金融时间序列具有高噪音、非平稳、非线性等特点，相比于传统机器学习算法，深度神经网络借助其深层的网络结构能够缓解传统方法易于陷入局部最优解的窘境。然而单纯的循环神经网络（Recurrent Neural Network，RNN）随着递归，会出现权重指数级爆炸或梯度消失的问题，因此难以捕捉长期时间关联，长短期记忆（Long-Short-Term Memory，LSTM）是一种 RNN 的变体，结合不同的 LSTM 可以很好的解决这个问题。

(三)研究内容及论文结构

1. 研究内容

本文研究利用 LSTM 的方法实现时间序列的预测，主要分析美国股票市场中的十家公司的日收盘价。本文的特色在于没有特意挑选具有一定趋势特征的某一家公司进行研究，而是从美国 1000 多家公司中随机选取 52 家并分析其中波动率较大的十家公司的收盘价格，这一操作大大提升了本次实验的挑战性。同时针对不同公司的金融数据，进行相对应的处理与建模分析，使得最终得到的模型更具有适用性。最后，本文将使用不同处理方式在十家公司上的表现结果以表格的形式清晰地进行展示，并对于未来进一步研究提出展望。

2. 论文结构

本文分为四章：

第一章为绪论，介绍本文的研究背景和意义并通过国内外文献综述介绍时间序列预测问题的研究现状；

第二章为理论阐述，对于传统统计学中的 ARIMA 模型，循环神经网络（RNN），长短期记忆（LSTM），门控神经单元（GRU）以及 LSTM 与 GRU 地对比进行阐述；

第三章为 LSTM 模型的应用，搜集美国股票市场的数据，并通过特征工程对于数据进行预处理，构建 LSTM 模型对于金融时间序列进行分析以及预测；

第四章为结论与展望，对于本文的主要工作进行总结并指出本文的不足之处以及在未来可实施的改进措施。

本文采用 Python 编程语言与 PyTorch 深度学习库进行 LSTM 的搭建、训练与预测。

理论阐述

(一)统计学方法

1. 相关概念

1)平稳性

在时间序列的研究中，序列的平稳性是至关重要的，是分析的基础。通常我们所说的平稳是指序列的弱平稳性，其定义为：

- 均值μ是与时间t无关的常数；
- 对于任何时刻 t 和任意的时间间隔 k，时间序列z_t与z_{t-1}的自协方差r_k只与时间间隔 k 有关，与 t 无关。

2)趋势

趋势是指时间序列在某一方向上的持续运动，是在较长时间内受到某种根本性因素的作用而形成的总的变动趋势。

3)季节变化

很多时间序列中包含着季节变化，它是在一年内随着季节的变化而发生的有规律的周期性变动。

4)序列相关性

序列相关性又称自相关性，是时间序列的一个最重要的特征，自相关性反应了时间序列中存在的规律，是可以预测未来的前提，没有了自相关性，该序列就成为了白噪声。

5)白噪声

白噪声的称呼来自于物理学，表示接收到的信号中只有噪声，没有信息。该序列的特点在于任何两个时间点的随机变量都不相关，即序列中没有可以利用的动态规律，因此，并不能通过历史数据对未来进行预测。

2. 自回归模型

自回归模型（Autoregressive model，AR 模型）是从回归分析中的线性回归发展而来，区别在于，AR 模型使用x_1和x_{t-1}预测x_t，即自身预测自身，并假设他们为线性关系，而非用 x 预测 y，所以称作自回归。p 阶 AR 模型数学表达式为：

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-1} + \varepsilon_t$$

其中，c 为常数项；ε_t是均值为 0，标准差为σ的随机误差项，且对于任何 t 均不变。文字表述为：X 的当期值为一个或数个前期值的线性组合，加上常数项与随机误差。

3. 移动平均模型

移动平均模型（Moving Average model，MA 模型）是对单一变量时间序列进行建模的方法，与 AR 模型不同，MA 模型总是平稳的，q 阶 MA 模型数学表达式为：

$$x_t = \mu + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q}$$

其中μ是序列的均值，θ是参数，ε是白噪声。

4. 自回归移动平均模型

上文所说的 AR 和 MA 模型“混合”构成自回归移动平均模型（Autoregressive Moving Average model，ARMA 模型），ARMA(p,q)包含了 p 个自回归项和 q 个移动平均项，它的数学表达式为：

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_1 X_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}$$

5. 差分整合移动平均自回归模型

差分整合移动平均自回归模型（Autoregressive Integrated Moving Average model，ARIMA 模型）是 ARMA(p,q)的扩展，非平稳序列经过 d 阶差分后被平稳化，并符合 ARMA(p,q)过程，则原序列记为 ARIMA(p,d,q)。

6. 具体预测的步骤

一般利用 ARIMA(p,d,q)模型进行时间序列预测有如下几步：

- 时间序列可视化并进行平稳性检验，非平稳序列通过差分将其平稳化，该步骤可确定 d 的取值；
- 确定 p 和 q 的值方法有很多：根据 PACF 和 ACF 图可以确定两个参数的值；也可以利用 BIC 或 AIC 为评判标准，以网格搜索（Grid Search）的方式找到使得 BIC 或 AIC 最小的 p 与 q 的组合值；在 Python 或 R 中，可以调用 auto arima 函数自动选出最佳的组合值；

- 生成模型报告，并进行预测。

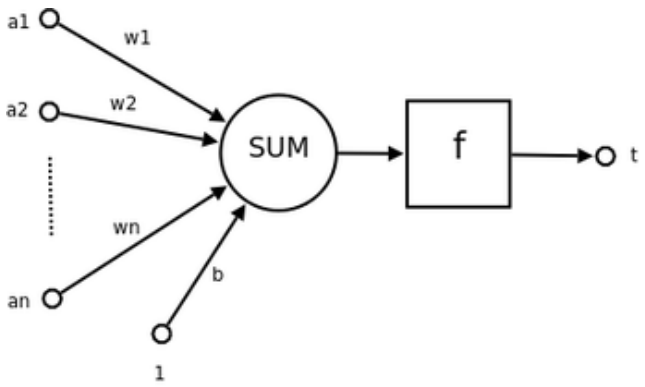
(二)深度学习方法

1. 神经网络

人工神经网络（Artificial Neural Network，ANN），简称神经网络（Neural Network，NN），它通过模仿生物学上神经网络的结构和功能生成模型，被广泛应用于机器学习和认知科学领域。通过大量的神经元联结并进行计算，实现对函数的估计或者近似。与传统的机器学习不同，神经网络具备较强的学习功能，可以自动地进行特征提取，不需要人为干预，是一种自适应系统。它作为一种建模工具，能够处理非线性数据，被用于解决如机器视觉和语音识别等的各种各样的问题。而这些问题通常是很难被传统基于规则的编程所解决的。

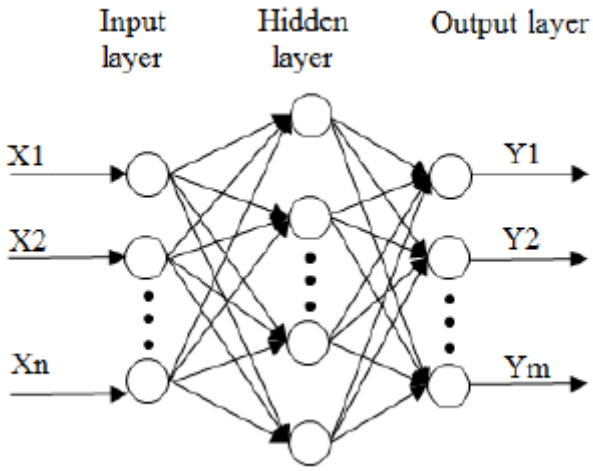
ANN 通常具有以下三个部分：

- 结构（Architecture）：这里的结构代指网络中的变量以及它们之间的关系。例如，神经元的激励值（Activities of Neurons）和神经元连接的权重（Weights）一般被看作是神经网络中的变量；
- 激励函数（Activation Function）：激励函数决定了一个神经节点的输出，通常依赖于网络中的权重（Weights）和神经节点的输入，激励函数的选取也可以说明该神经元是如何根据其他神经元的活动来调整激励值的，常用的激励函数包括：ReLu，tanh，Softmax，Sigmoid 等；
- 学习规则（Learning Rule）：它是一个迭代的学习过程，具有数学逻辑性。当神经网络在一个特定的数据环境下进行模拟时，学习规则通过不断更新神经网络的权重（Weights）与偏差（Bias）从而提高其最终的表现，常见的学习规则有：Hebb 学习规则，感知机学习规则，最小均方规则等等。



常见的多层结构的前馈网络（Multilayer Feedforward Network）通常由三部分组成：

- 输入层（Input layer）：外界信息通过输入层进入神经网络；
- 输出层（Output layer）：输入信息在神经网络中经过一系列的操作最终通过输出层输出结果；
- 隐藏层（Hidden layer）：简称“隐层”，它位于输入层和输出层之间，有众多神经元和链接组成，是神经网络中最重要的部分。对于输入信息进行的一系列操作大多都是在隐层进行。它可以有一层或多层，且节点的数目不确定，但一般来说节点数目越多，神经网络的非线性越显著，从而神经网络的鲁棒性（robustness）越强，隐藏层节点数一般会选输入节点数的 1.5 倍。



人工神经网络依据网络架构分类主要有：

- 前馈神经网络（Feed Forward Network）
- 循环神经网络（Recurrent Network）

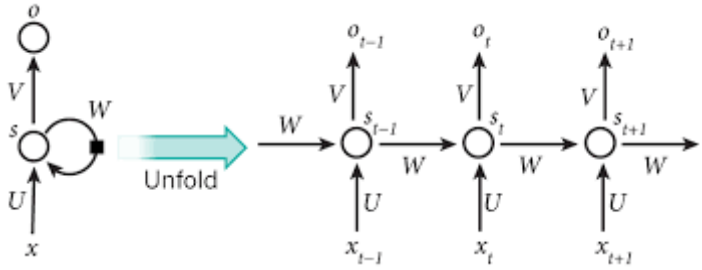
- 强化式架构（Reinforcement Network）

本文主要利用循环神经网络（Recurrent Network）来预测时间序列。

2. 循环神经网络

传统的神经网络只能单独的处理一个个的输入项，前一个输入项与后一个是独立的，然而现实生活中的很多数据具有时间上的相关性，尤其是时间序列。循环神经网络（Recurrent Neural Network，RNN）可以将输入项处理成序列形式，从而在计算输出项的时候，该时间点之前的信息也可以被考虑到。RNN 通过这种方式使得神经网络能够利用不同时间步的信息，从而解决了将所有的输入项当作互不相关的信息处理所产生的问题。

RNN 的一般形式和展开形式如下图：



RNN 由输入层、隐藏层和输出层组成。而隐藏层不仅与输出层相连接，同时与隐藏层自身也有连接，这说明隐藏层的值不仅取决于当前时刻的输入，还取决于上一时刻隐藏层的输出。U 为输入层到隐藏层的的权重矩阵，而 W 则是隐藏层的上一时刻的值作为当前时刻的输入的权重矩阵。用数学表达式说明如下：

$$h_t = f(Ux_t + Wh_{t-1})$$
$$o_t = g(Vh_t)$$

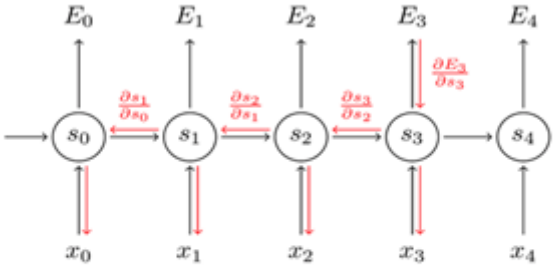
其中 f 和g分别为隐藏层与输出层的激活函数，h_t 和o_t为隐藏层和输出层在t时刻的值。

在使用单纯的 RNN 时，随着递归，容易出现梯度爆炸或者梯度消失的问题，进而导致难以捕捉长期的时间关联性。

关于梯度爆炸和梯度消失的问题，在此给出解释：

神经网络的训练一般分为前向传播与反向传播，前向传播如上文中所述，RNN 的反向传播 BPTT（Back Propagation Through Time)的基本原理同 BP 算法一致，根据损失函数计算误差并利用梯度下降算法通过梯度的反向传播，实现网

络参数，主要是各个层的权重矩阵，的更新优化。如下图所示：



Backpropagation Through Time

对于$t=3$, U、W、V求偏导

$$\frac{\partial L_3}{\partial V}=\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial V}$$

$$\frac{\partial L_3}{\partial W}=\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\frac{\partial s_3}{\partial W}+\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_2}\frac{\partial s_2}{\partial W}+\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial W}$$

$$\frac{\partial L_3}{\partial U}=\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\frac{\partial s_3}{\partial U}+\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_2}\frac{\partial s_2}{\partial U}+\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial U}$$

对U、W的偏导可以简写为

$$\frac{\partial L_3}{\partial U}=\sum_{k=0}^{3}\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\left(\prod_{j=k-1}^{3}\frac{\partial s_j}{\partial s_{j-1}}\right)\frac{\partial s_k}{\partial U}$$

$$\frac{\partial L_3}{\partial U}=\sum_{k=0}^{3}\frac{\partial L_3}{\partial o_3}\frac{\partial o_3}{\partial s_3}\left(\prod_{j=k-1}^{3}\frac{\partial s_j}{\partial s_{j-1}}\right)\frac{\partial s_k}{\partial U}$$

注意到其中累乘的部分，$\frac{\partial s_j}{\partial s_{j-1}}=f'W$,当激活函数是 tanh 或 sigmoid 时（tanh 和 sigmoid 的导数值最大分别为 1 和 1/4），大多数是小于 1 的数值在做累乘，当 t 很大时，导数累乘的部分将趋于 0，这便是 RNN 中梯度消失的原因。当网络参数 W 的数值过大，$\tanh'W>1$，随着序列长度存在长期依赖的问题，就会产生梯度爆炸。在实际运用中，RNN 的神经网络层数较多，这就使得梯度消失或梯度爆炸问题较为明显。

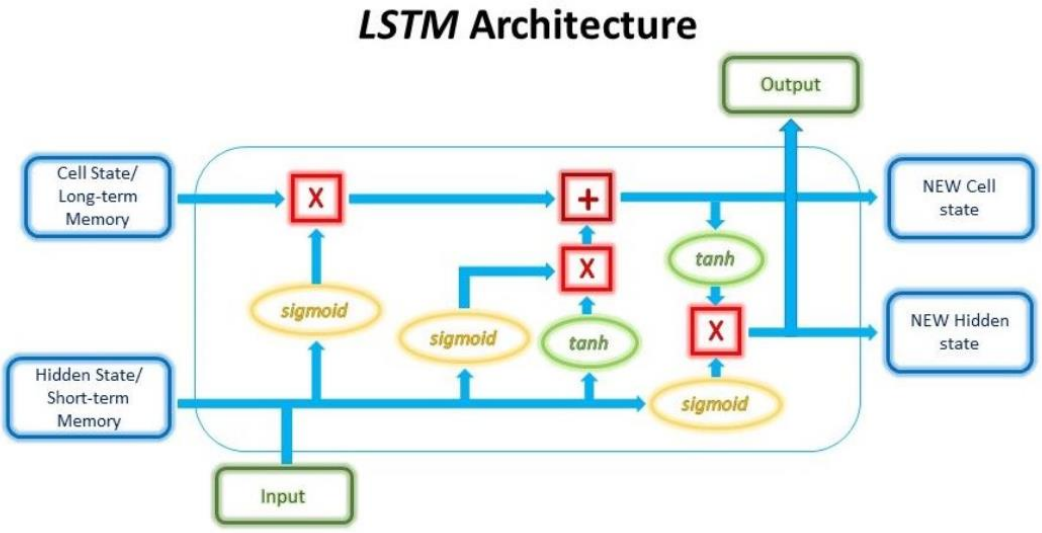
对于梯度爆炸，可以使用一些调参的技巧。而为了减少梯度消失所带来的不利影响，可以选择 ReLu 作为激活函数，或者改变神经网络的内部结构，也就是接下来所说的长短期记忆（LSTM）与门控循环单元（GRU）。

3. 长短期记忆

LSTM 的基本单元包括三个控制门：遗忘门（Forget Gate）、输入门（Input

Gate）和输出门（Output Gate）以及一个记忆细胞单元 cell。在每个时刻，LSTM 考虑三部分的信息：当前的输入信息X_t，来自上一个单元的短期记忆h_{t-1}（类似于 RNN 中的隐藏层信息）和来自记忆细胞单元 cell 的长期记忆c_{t-1}。

LSTM 进行状态更新和输出信息的过程如下：首先输入门根据输入和上一时刻输出进行状态的更新，遗忘门控制保留有用历史信息的比例，Cell 单元根据当前输入、上一时刻输出、历史记忆 t−1 对 cell 状态进行更新，最后在输出门的控制之下，输出当前时刻的信息h_t。



Inner Workings of the LSTM cell

下面对于具体流程以及三个控制门进行详细说明：

1. 输入门：

输入门决定了什么样的信息可以存储在长期记忆中，它仅作用于当前的输入X_t以及来自上一时刻的短期记忆h_{t-1}，输入门负责过滤掉这些信息中无效的部分。

输入门一般有两层，第一层类似于一个过滤器，过滤掉无用信息，这一层的创建是利用 sigmoid 函数，该函数会将短期记忆h_{t-1}以及当前输入X_t这两部分信息转化为 0~1，0 代表这部分信息完全不被利用，1 代表这些信息能够被利用。通过这种方式，可以实现信息的过滤：

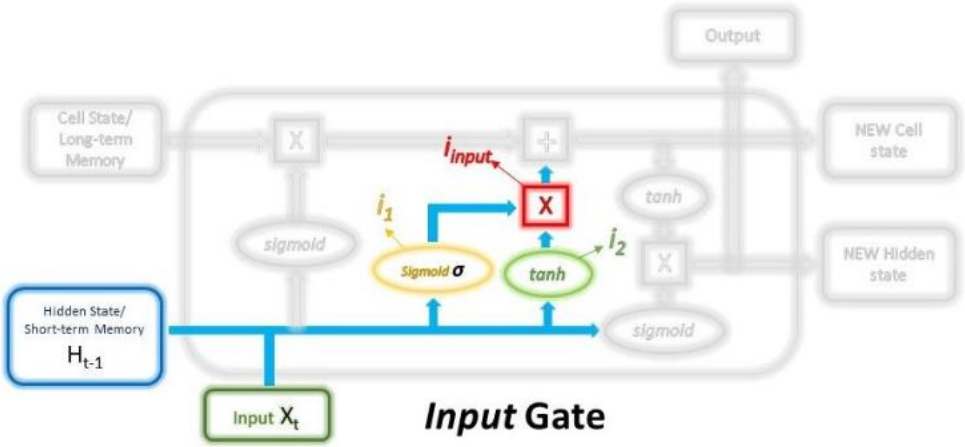
$$i_1 = \sigma\left(W_{i_1} \cdot (H_{t-1}, x_t) + bias_{i_1}\right)$$

第二层将通过将短期记忆h_{t-1}以及当前输入X_t放入激活函数，通常为 tanh，来调节神经网络。最后将这两层的输出相乘作为输入门最终的输出并与遗忘门的

输出相加作为新的长期记忆c_t:

$$i_2 = \tanh(W_{i_2} \cdot (H_{t-1}, x_t) + bias_{i_2})$$

$$i_{input} = i_1 * i_2$$



Input Gate computations

2. 遗忘门：

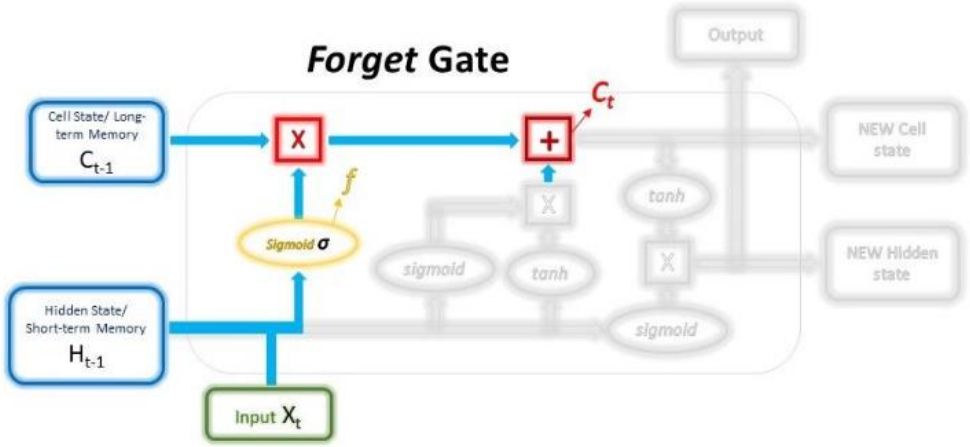
遗忘门用来过滤来自长期记忆c_{t-1}的信息，通过将长期记忆c_{t-1}乘以一个由当前输入X_t与短期记忆h_{t-1}生成的遗忘向量（forget vector）实现。

和输入门中的第一层类似，遗忘向量如同一个过滤层，通过将短期记忆h_{t-1}与当前输入X_t放入 Sigmoid 的函数中来获得。遗忘向量与长期记忆相乘的结果表示最终长期记忆中的哪些信息将被利用:

$$f = \sigma\left(W_{forget} \cdot (H_{t-1}, x_t) + bias_{forget}\right)$$

输入门和遗忘门的输出做逐项相加的结果作为新的长期记忆c_t进入下一单元，同时被用于输出门:

$$C_t = C_{t-1} * f + i_{input}$$



Forget Gate Flow

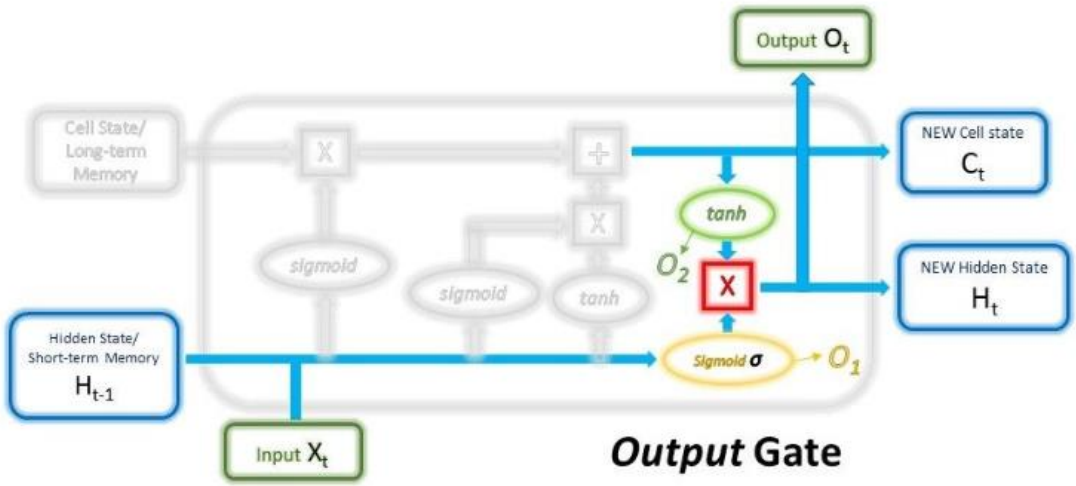
3. 输出门：

首先，和输入门与遗忘门相似，之前的短期记忆h_{t-1}与当前输入X_t被放入 Sigmoid 函数中（每次的权值矩阵是不同的）来得到过滤器。之后，我们将新的长期记忆c_t通过激活函数 tanh 的结果与上一步得到的过滤器相乘从而产生新的短期记忆h_t:

$$O_t = \sigma\left(W_{output1} \cdot (H_{t-1}, x_t) + bias_{output1}\right)$$

$$O_2 = \tanh\left(W_{output2} \cdot C_t + bias_{output2}\right)$$

$$H_t, O_t = O_1 * O_2$$



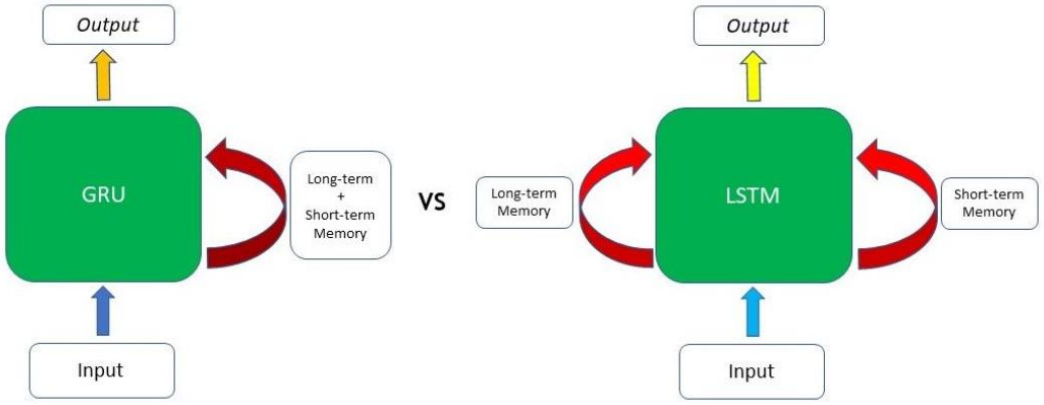
Output Gate computations

由这三个门控结构生成的新的短期记忆h_t和长期记忆c_t将被传递到下一个单元，进行相同的操作。短期记忆h_t即为 LSTM 单元每一个时刻的输出。

4. 门控循环单元

门控循环单元（Gated Recurrent Unit，GRU）和 LSTM 类似，都是 RNN 的一种变体，可以有效地获取时间序列数据中的长期记忆信息，从而解决普通循环神经网络只有短期记忆的窘境。

LSTM 于 1997 年被提出，而 GRU 在 2014 年被 Cho 提出，GRU 与 LSTM 都是利用其门控结构实现长期记忆的获取，不同的地方在于：LSTM 有两个层结构分别控制长期记忆与短期记忆在时间步之间的传递，而 GRU 只有隐藏层同时传递长期和短期记忆。



GRU vs LSTM

比起 LSTM 拥有三个门控结构，GRU 只有两个门：更新门（update gate）和重置门（reset gate）。与 LSTM 类似，这两个门通过被训练，成为包含了数值在 0 到 1 的元素的向量，与输入数据或者隐藏层相乘，实现对于不相干信息的过滤及有用信息的提取。

GRU 的实现主要分为以下三步：

1. 重置门：

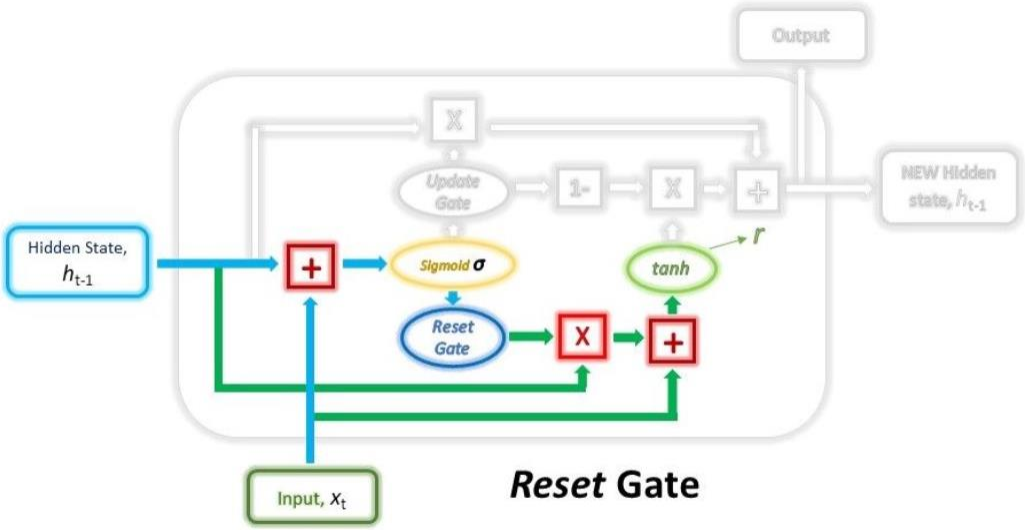
第一步，构建重置门，它是通过上一时间点的隐藏层h_{t-1}和当前时间点的输入项X_t的计算得到。通过对于这两项赋予权重后相加的结果输入 sigmoid 函数中，我们可以得到处于 0 到 1 之间的值，作为过滤向量：

$$gate_{reset} = \sigma\left(W_{input_{reset}} \cdot x_t + W_{hidden_{reset}} \cdot h_{t-1}\right)$$

当整个神经网络通过反向传播训练，这里的权值将会被不断地更新，最终过滤向量实现仅保留有效信息的功能。

同时，我们对于上一时间点的隐藏层赋予权重，并与上文中得到的过滤向量进行逐元素相乘，这一步实现了对于前一个时间点的信息的过滤。同时，当前的输入项也会被赋予一个权重并与刚刚所得结果相加，一同放入非线性激活函数 tanh 中，从而得到重置门最终的输出：

$$\mathrm{r} = \tanh\left(gate_{reset} \odot \left(W_{h_1} \cdot h_{t-1}\right) + W_{x_1} \cdot x_t\right)$$



Reset Gate Flow

2. 更新门：

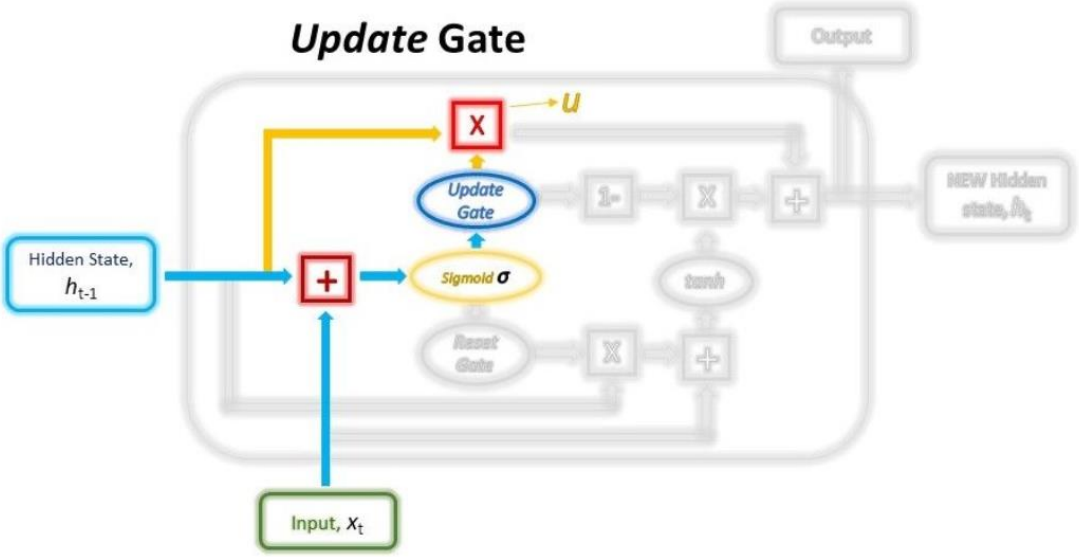
第二步，构建更新门。和重置门相似，更新门也是通过上一时间点的隐藏层和当前输入数据计算得到。两个门控单元的过滤向量的计算方式是一样的，注意这里的权重是不一样的：

$$\mathrm{gate_{update}} = \sigma\left(\mathrm{W_{input_{update}}} \cdot \mathrm{x_t} + \mathrm{W_{hidden_{update}}} \cdot \mathrm{h_{t-1}}\right)$$

更新门的过滤向量之后将与前一个时间点的隐藏层进行逐元素相乘，得到的结果将用于最后一步的输出：

$$\mathrm{u} = \mathrm{gate_{update}} \odot \mathrm{h_{t-1}}$$

更新门的过滤向量同时也会被用在最后一步。更新门的作用就是帮助模型筛选出储存在上一个时间点的过去信息中需要被保留的部分，用于之后的操作。



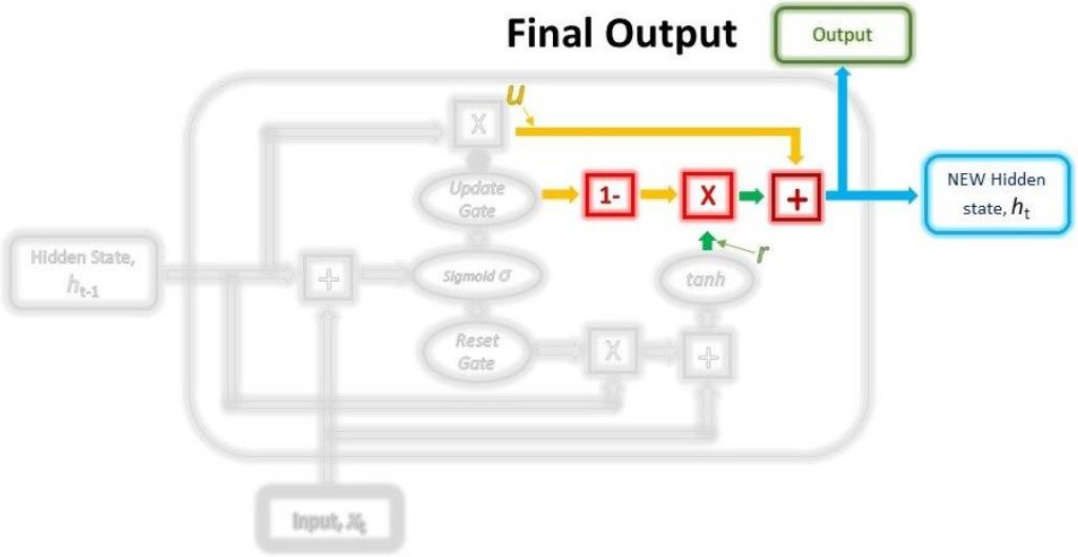
Update Gate Flow

3. 结合输出

最后一步，我们将利用第二步的更新门得到更新后的隐藏层$\mathrm{h_t}$，并作为该单元的输出。

首先，我们将 1 减去更新门的过滤向量$\mathrm{gate_{update}}$。然后，将结果与重置门的输出 r 进行逐元素相乘，这一步筛选出了新信息中将被储存在更新后的隐藏层中的部分。最后，将上一步的结果与更新门的输出 u 相加，作为新的更新后的隐藏层，同时也是当前时间点的 GRU 单元的输出：

$$\mathrm{h_t = r \odot \left(1 - gate_{update}\right) + u}$$



Final Output Computations

5. LSTM 与 GRU 的对比

下面从结构、训练速度以及模型表现三个方面比较 LSTM 与 GRU：

- 结构上：LSTM 与 GRU 都具有门控结构，但是 LSTM 有三个门，GRU 只有两个。GRU 的更新门和 LSTM 中的输入门与遗忘门的功能相似。但是，在对当前时间点的新信息的控制上，两者有所不同。LSTM 中，遗忘门负责筛选长期记忆c_{t-1}中的有用信息，而输入门则决定了当前输入项X_t与短期记忆h_{t-1}中所要保留的部分。这两个门是彼此独立的。而对于 GRU，由于更新门兼具了以上两个功能，所以，这两部分信息的筛选与处理并不独立。另一个不同在于：LSTM 将长期与短期记忆独立的存放在两个层中，而 GRU 则将它们储存在一个隐藏层里。
- 训练速度：由于 GRU 的权重矩阵与其他参数比 LSTM 更少，所以在训练时，GRU 的速度会比 LSTM 要快。
- 模型表现评估：GRU 与 LSTM 均为 RNN 的变种，表现都可以达到比较理想的水平，模型的准确率一般而言 LSTM 会比 GRU 稍高，但考虑到其耗时更长，对于具体问题，应该具体判断选择哪种模型。

基于 LSTM 的时间序列预测实例

(一)数据获取与描述

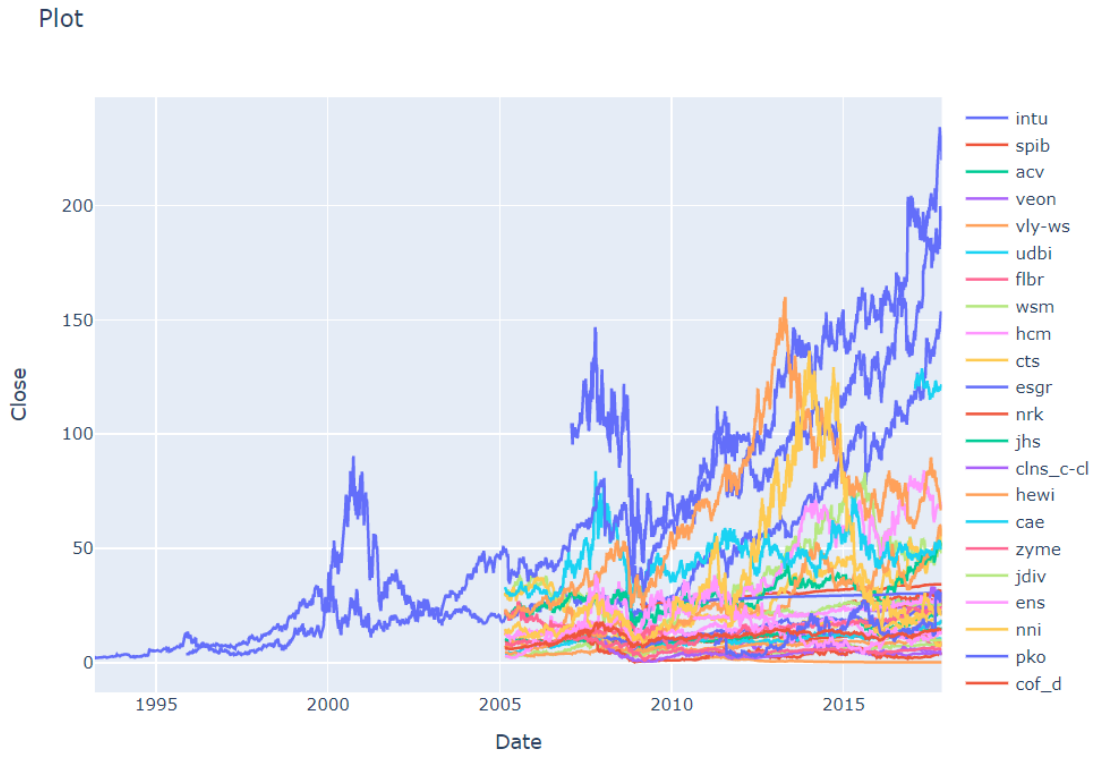
本次数据来源于 Kaggle 的分享者的整理汇总，整个数据集中包含了美国1000+家公司的股票数据（截止至 2017 年 11 月 10 日），以.csv 格式保存。

每家公司的数据包括以下特征：

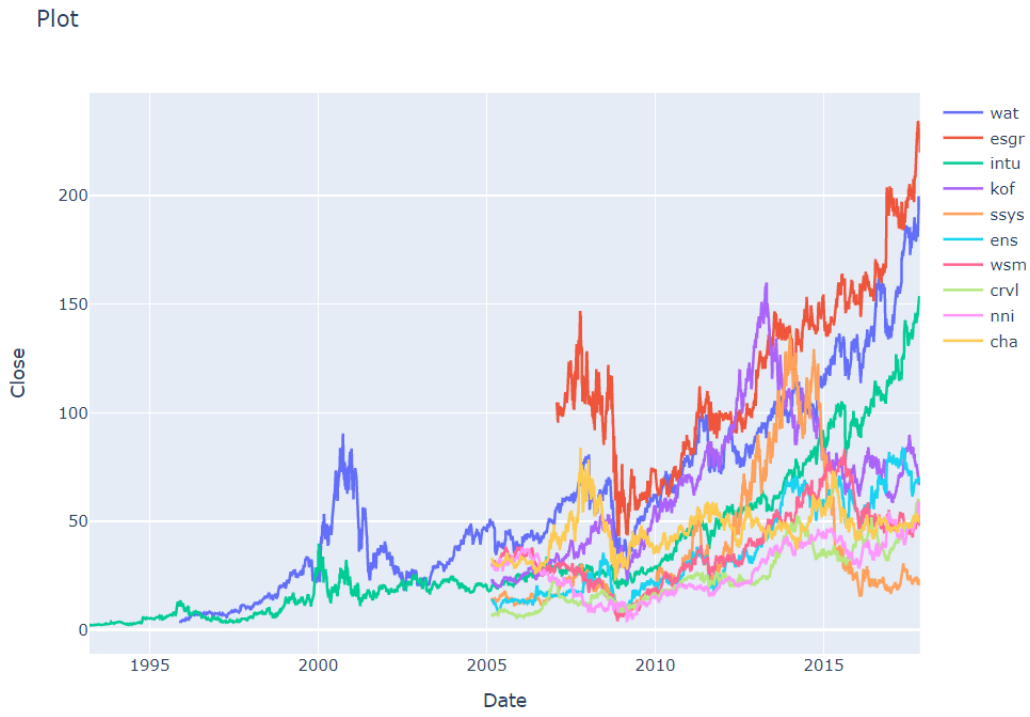
- Date: yy-mm-dd，股票交易日期，一般按照工作日；
- Open：开盘价，即某种证券在证券交易所每个交易日开市后的第一笔买卖成交价格；
- High：最高价，即某种证券在证券交易所每个交易日开市后的最高成交价格；
- Low：最低价，即某种证券在证券交易所每个交易日开市后的最高成交价格；
- Close：收盘价，即某种证券在证券交易所每个交易活动结束前的最后一笔买卖成交价格，收盘价被看作股市当天行情的标准，也是下一个交易日开盘价的依据，所以在分析行情时，常用收盘价作为计算依据；
- Volume：市场成交量，的变化反映了资金进出市场的情况，用于印证市场走势；
- OpenInt：持仓量，当天收市时的所有以流通的或未平仓的合约总数，代表市场中多头或空头的流通流通合约总数，而非两者之和，在本数据集中为 0。

(二)数据展示

考虑到整个数据集较大，无法同时对于 1000+公司数据进行处理，在获取数据的时候，随机选取了 52 家公司，并将他们的收盘价（Close）绘制如下：



可以看出，52 家公司部分起伏较大，部分较为平稳，数学上来看，起伏较大的公司收盘价的标准差较大，平稳的公司标准差较小。对于起伏较大的公司，研究其收盘价更具挑战性，故最终挑选出这 52 家公司中收盘价标准差较大的十家公司作为研究对象：



(三)特征工程

通常来说，机器学习或深度学习方法的上限是由数据本身的特性，如数据量的大小和数据特征的数量，决定的，而选用的模型和算法影响的是逼近这个上限的效果。

特征工程目的就是最大限度地从原始数据中提取合适的特征以供算法和模型使用，是机器学习或深度学习过程中必不可少的一步，它是打开数据密码的钥匙，也是数据科学中最有创造力的一部分。因为往往和具体的数据相结合，所以做好特征工程的基础是对于处理的数据有一个清楚的认识。

对于本次数据，原数据中的特征只有 6 个，显然特征数量很少，故特征工程的重点在于结合统计学和金融学的知识，利用原始数据，构造出更多特征，并选取出能够提高模型效果的部分特征。

1. 缺失值检验

利用 Python 中的.isnull()即可检验是否含有缺失值，此数据较为干净，不存在缺失值或异常值。

2. 构造新特征

本次新特征的构造主要是利用统计学中的常见统计量与金融学中股票市场的技术指标，下面对于添加的新特征进行说明：

1) 统计学上：

通过对于研究对象日收盘价提取其历史信息，并利用滑动窗口（rolling window）的方式获取其历史标准差，均值，最大值，最小值，窗口大小的可设置为 5、10、21、63、126（分别为一周、两周、一个月、一个季度、半年的工作日时长），最多可添加 20 个统计学新特征。

2)金融学上：

对于股票市场，有很多可供参考的技术指标（Technical Indicators），技术指标主要分为下面四大类：

- 动量：现金流指数（MFI）、相对强弱指数（RSI）、真实强弱指数（TSI）等；
- 市场趋势：顺势指标（CCI）、指数平滑移动平均线（MACD）、移动平均（MA）等；

- 交易量：强力指数（FI）、能量潮（OBV）、卖权-买权比率（PCR）、量价曲线（VPT）等；
- 波动性：真实波动幅度均值（ATR）、股市不稳定指数（VIX）、标准差（σ）等 。

当我们利用其中一个或多个指标时，要保证他们来自不同的类别，否则将会出现信息冗余的情况。

本次实验利用以下几个常用的技术指标构建新特征组合：

- 相对强度指数（Relative Strength Index，RSI）：

RSI 于 1978 年被提出，是一种动量指标，用于衡量近期价格变化的幅度，从而评估股票或者其他资产价格中的超买或者超卖的情况。RSI 由一段时间内市场的涨跌比率制作而成的曲线，是在两个极端之间移动的折线图。

RSI 的计算方式：分别计算出一段周期内的上涨对应收盘价的涨幅平均值和下跌对应收盘价的跌幅平均值，并计算涨幅占整体波动幅度的比率。一般情况下，RSI 超过 70%被看作超买，低于 30%被看作超卖。

- 平滑异同移动平均线（Moving Average Convergence and Divergence, MACD）：

MACD 于 1970 年代被提出，一种趋向类指标。它通过对价格收盘价进行指数滑动平均（Exponential Moving Average，EMA 或 EXMA）处理后计算得到的，用于判断股票价格变化情况，以便把握股票买进和卖出的时机。

MACD 的计算公式为 MACD= (12-day EMA - 26-day EMA)：

其中 EMA 是以指数式递减加权的移动平均。各数值的加权影响力随时间而指数式递减，时间上越近的数据权值越重。加权的程度以常数 α 决定，α 数值介乎 0 至 1。α 也可用天数 N 来代表：$\alpha = \frac{2}{N+1}$，EMA 的计算可以通过 Python 中的.ewm()并设置 period=12 与 26 得到。MACD 线的上方会绘制 9-day EMA，它被称为“信号线”，当 MACD 穿越其上方，发出买入信号，穿越下方，交易者可进行卖出或做空操作。

- 能量潮（On Balance Volume，OBV）：

OBV 指标于 20 世纪 60 年代被提出的，并被广泛使用，OBV 将股市的成交量变化作为股价走势的评判标准。其计算公式为：

$$\mathrm{OBV_t} = \mathrm{OBV_{t-1}} + \mathrm{sgn} \times \mathrm{Volume_t}。$$

其中，sgn是符号函数，其数值由下式决定：

$\mathrm{sgn} = +1$ 今日收盘价>昨日收盘价；

$\mathrm{sgn} = -1$ 今日收盘价<昨日收盘价；

$\mathrm{sgn} = 0$ 今日收盘价=昨日收盘价。

3. 特征缩放

对于特征量纲差别较大的情况，我们需要通过特征缩放使得特征的量级保持一致。特征缩放的好处在于：能够使得梯度下降算法更快的收敛。特征缩放可以利用 sklearn 中的 preprocessing 实现，常用的方式有：

- 最值缩放：利用该特征的最大最小值将其缩放到[-1,1] 或[0,1]的范围；
- 标准化：利用该特征的均值和标准差进行缩放，处理后的数据均值为 0，标准差为 1；
- Log 缩放：对于数据取对数。

本次所使用的缩放方式为最值缩放，可以利用 sklearn.preprocessing.MinMaxScaler()实现。

这里需要注意：通常情况下，如果特征数不止一个，且他们的量纲差别较大，需要进行缩放处理。而类似地，如果预测的目标 y，不是一维，譬如输出是一个三维向量，且每一维度的量纲也差别较大，在处理时，不仅特征需要进行缩放，目标也需要进行缩放处理。但是，在本次实验中，虽然预测的目标：日收盘价是一维的数据，如果在训练前不进行缩放处理，最终预测出的结果会被一个阈值所限制。对此现象，目前还没有合理的解释，需要进行后续的研究。

4. 数据维数设置

对于 LSTM，输入的数据维度为三维，分别对应着：样本数目，时间步与特征数目，同时，需要将数据转化为 Tensor 的形式。

(四)划分训练集与测试集

对于已经处理好的数据，将其按照 7：3 的比例划分训练集与测试集。

训练集的特征与目标用于训练模型，将测试集的特征输入到最终训练出的模型得出预测结果，并与实际情况相对比。

如果需要进行调整参数，可以按照 7：1：2 的比例划分训练集、验证集与测

试集：其中验证集用于参数调整的过程。

如果需要增加训练集样本数，则对于其余九家公司的数据做前 4 步相同的处理后，选出日期在原本训练集最后一天之前的所有数据，将他们合并构造出新的、样本数更多的训练集。

(五)搭建神经网络

本次 LSTM 的搭建利用 PyTorch 中的 nn 实现，其中需要设定的内容有：

- input_size：输入的特征维数，根据添加特征的数量而定；
- hidden_size：自定义，通常为 2 的指数次幂，本次使用 64；
- num_layers：神经网络的层数，通常使用 1 或 2，本次使用 2 层。

(六)训练

1. 训练集

经过反复实验，对于不同的公司，最终确定不同的训练集：

- 对于标准差较小的公司而言，训练集的部分只需要本公司的数据就能训练出预测效果很好的模型；
- 而对于标准差较大的公司（前三名的公司）而言，仅仅使用本公司的训练集所得到的模型并不能够很好的对测试集进行预测，即泛化能力差，属于欠拟合问题。可以通过前文提到的增加样本数据量的方式提高其泛化能力。

2. 过拟合与欠拟合问题

下面对于机器学习和深度学习中常见的过拟合与欠拟合问题做出解释：

泛化能力是指算法对于新鲜样本的适应能力，在本次实验中可以看作是训练后的 LSTM 模型在测试集上的表现能力，而过拟合和欠拟合是导致模型泛化能力不高的两种原因。“欠拟合”通常是因为模型的学习能力弱，没能学习到数据集中的一般规律。相对应的，“过拟合”则是因为模型的学习能力太强，将某些样本自身特有的特征看作是数据集的一般规律。通常机器学习算法有“偏差-方差平衡”，最优的情况是偏差和方差都处于较小的位置，而欠拟合往往表现为高偏差、低方差，过拟合为低偏差、高方差。

过拟合出现的原因较为多样，解决方式也更为繁琐，通常有以下方法：

- 在神经网络中，使用权值衰退的方式，每次迭代过程以某个小因子来降低每个权值；
- 通过增加验证集对于模型参数进行调整；
- 进行交叉验证；
- 正则化：在目标函数后面加上正则项，通常有岭回归，LASSO 等方式。

比起过拟合，欠拟合问题的解决更为简单，可以通过如下方式：

- 增加新特征：通过加入特征组合、高次特征等；
- 增加多项式特征：例如在线性模型中加入二次项或高次项；
- 增加训练集样本数：通过更多的样本使得模型的训练更加充分，对于神经网络而言，较大的样本数是训练模型的重要因素之一；
- 减少正则化参数：正则化是为了防止过拟合，当模型出现欠拟合时，应该减少正则化；
- 集成学习方法：将多个弱学习器集成来提高模型的泛化能力。

3. 批训练

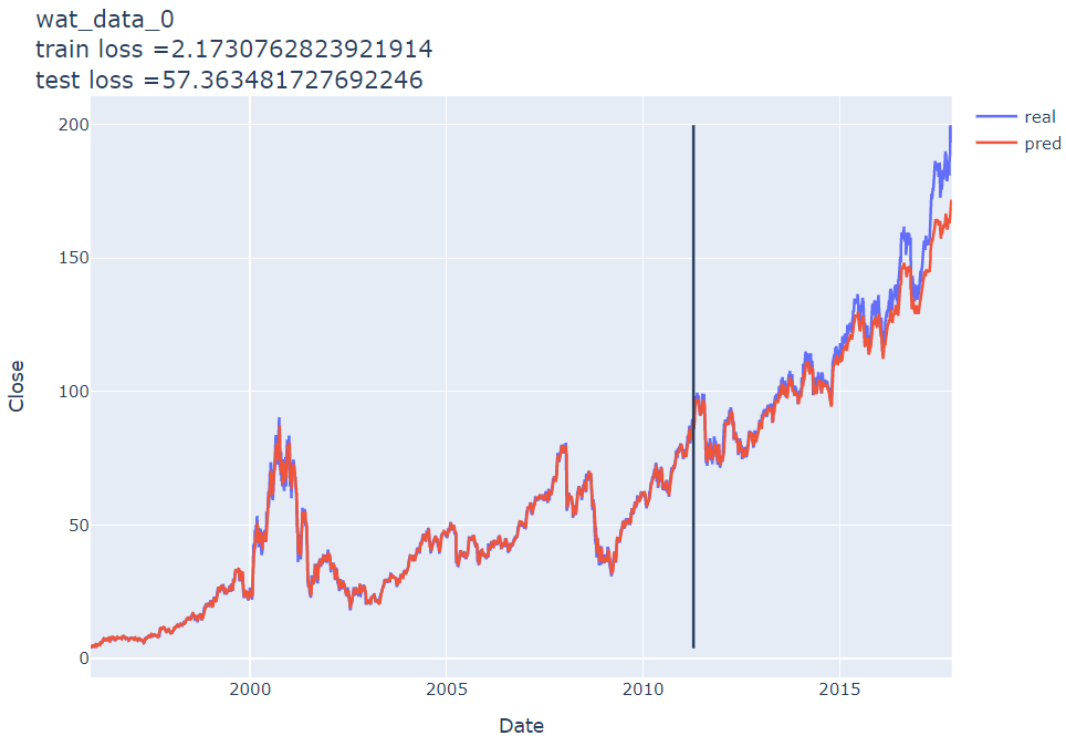
当我们使用样本量较大的训练集时，训练的时间明显增加，为了在利用较大的训练集、保证泛化能力的同时减少训练耗时，我们考虑使用批训练（Batch Training）的方式。batch_size 一般取 2 的指数次幂，本次取 2 的 8 次幂 256（和一年中的工作日时长近似）。

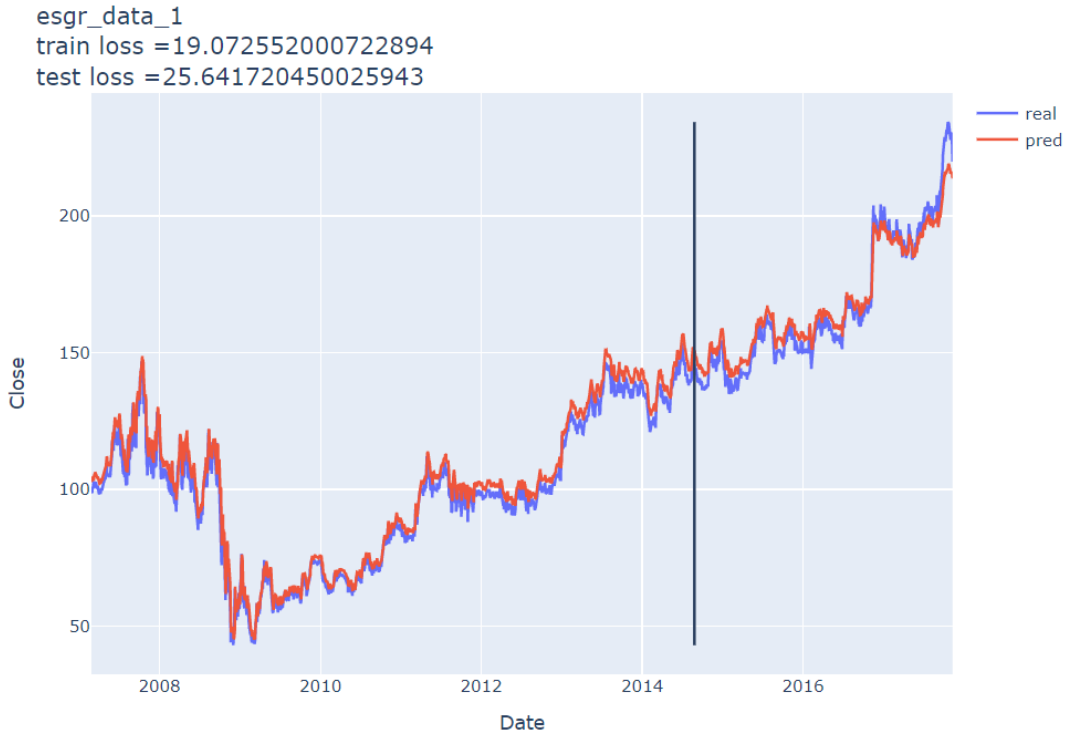
(七)预测

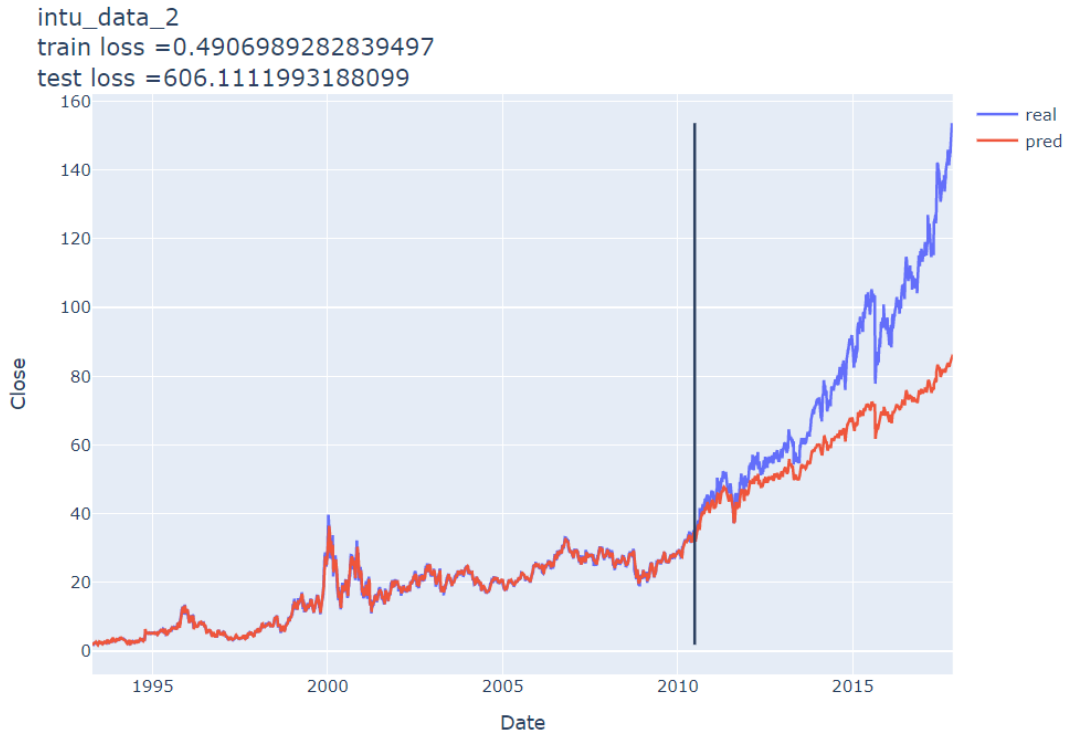
下面对于预测结果进行展示，由于共有十家公司，每家公司利用 2 种训练集，尝试了 6 种不同的特征组合，最终的成果（以图的形式展示）无法逐一放入论文中进行分析，故仅放出具有代表性的公司的预测情况，其他结果见附录。

1. 只利用原有特征

以下是这十家公司只利用原有的特征且只利用本公司的训练集进行模型训练后在测试集上的预测结果展示，其中左上角为公司名称及该公司的标准差排名（0~9），训练集和测试集的损失（MSE），蓝线为真实值，红线为预测值，黑色竖线左右分别为训练集与测试集：

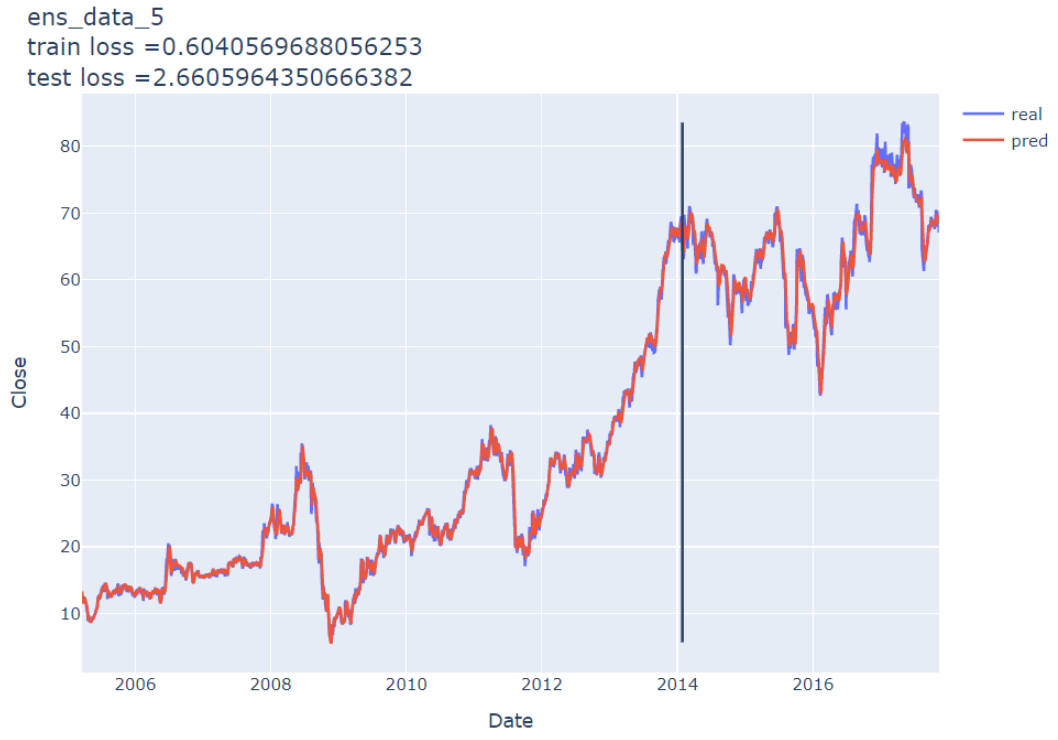

wat_data_0
train loss =2.1730762823921914
test loss =57.363481727692246
real
pred
Close
Date
200
150
100
50
0
2000
2005
2010
2015


esgr_data_1
train loss =19.072552000722894
test loss =25.641720450025943
real
pred
Close
Date
200
150
100
50
2008
2010
2012
2014
2016

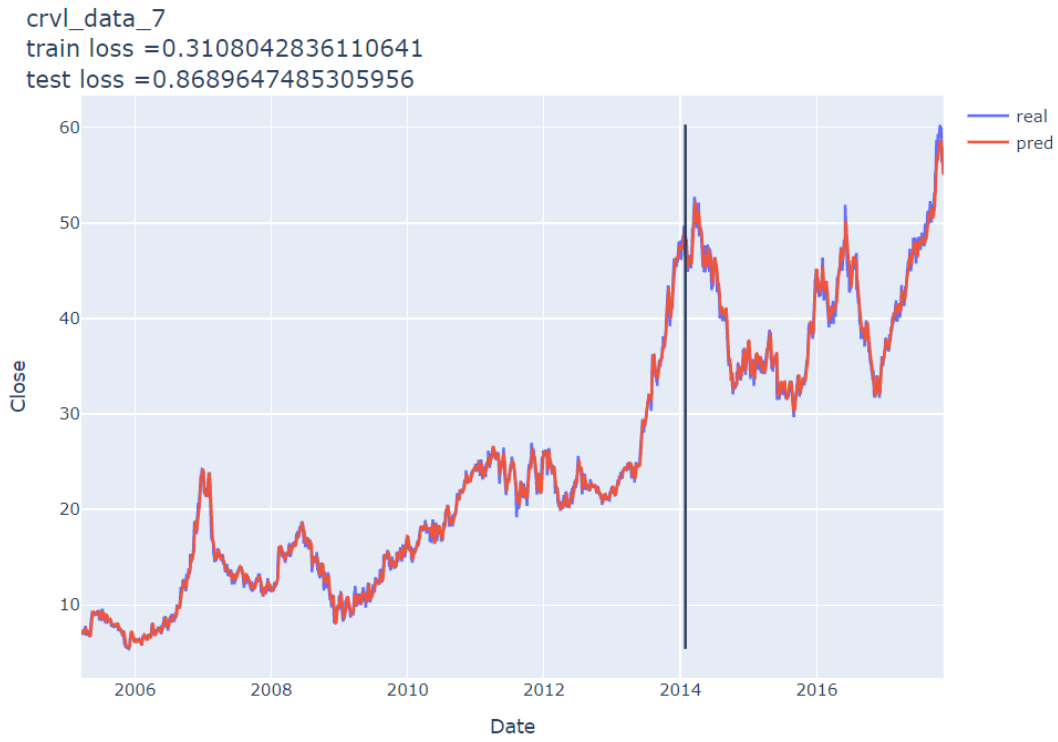

intu_data_2
train loss =0.4906989282839497
test loss =606.1111993188099
real
pred
Close
Date
0
20
40
60
80
100
120
140
160
1995
2000
2005
2010
2015

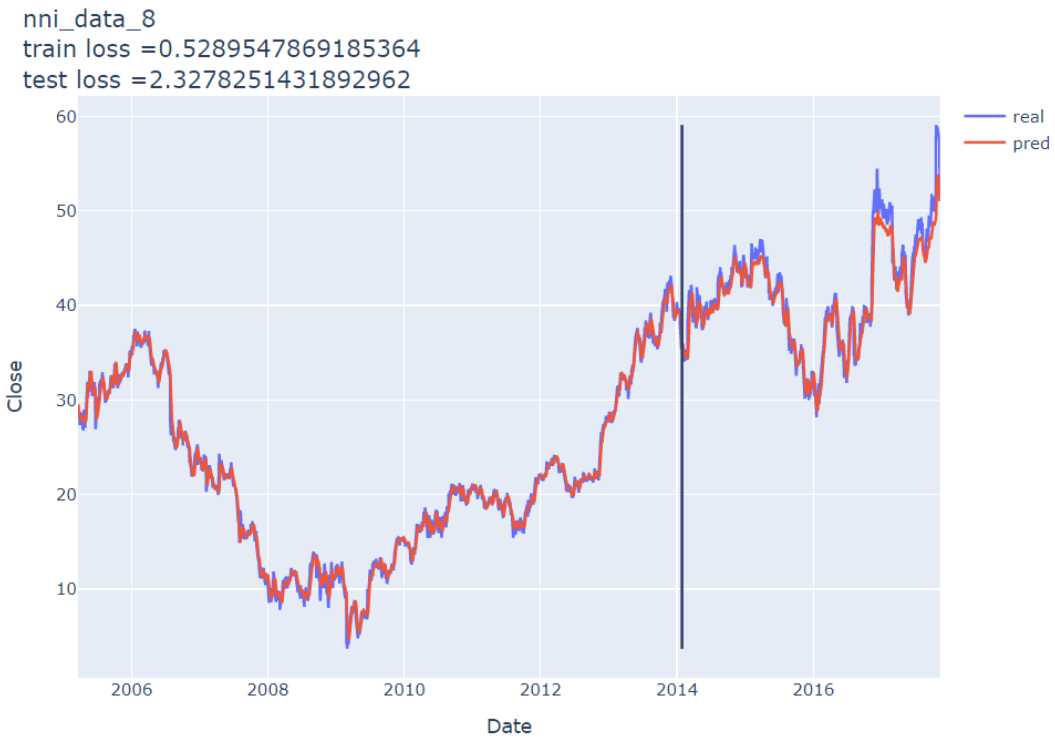

kof_data_3
train loss =2.518260748744676
test loss =2.5933895758137098
real
pred
Close
Date
20
40
60
80
100
120
140
160
2006
2008
2010
2012
2014
2016

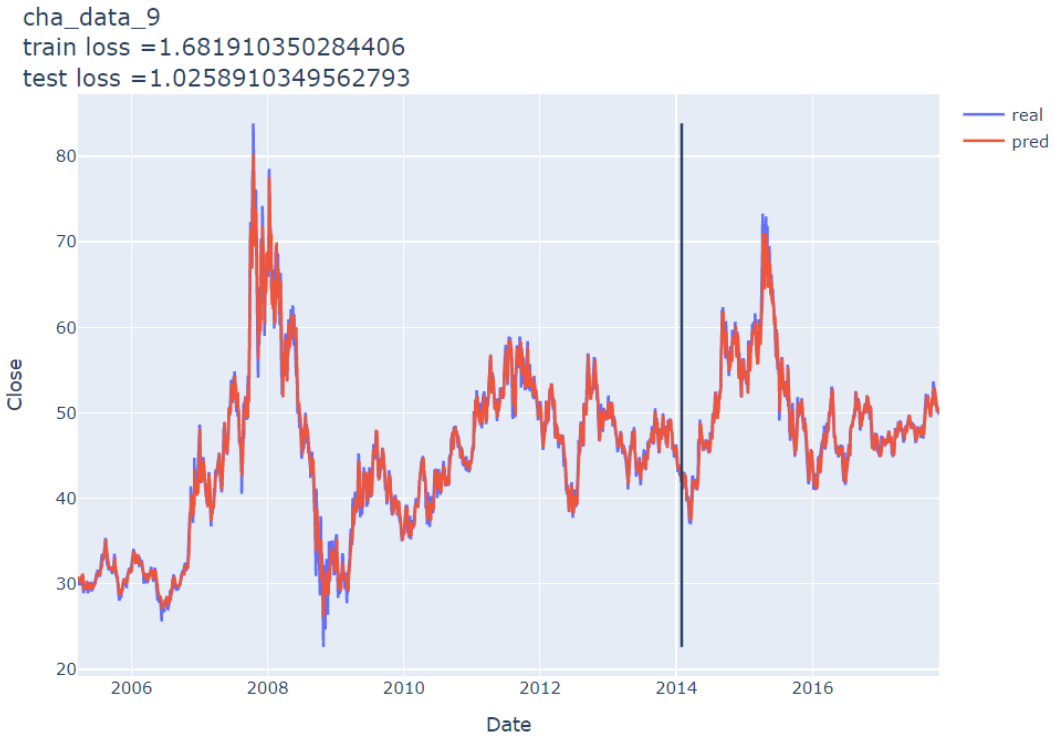

ssys_data_4
train loss =3.2473908141371526
test loss =5.776138227784374
real
pred
Close
Date
140
120
100
80
60
40
20
2006
2008
2010
2012
2014
2016


ens_data_5
train loss =0.6040569688056253
test loss =2.6605964350666382
real
pred
Close
Date
80
70
60
50
40
30
20
10
2006
2008
2010
2012
2014
2016


wsm_data_6
train loss =0.7629114458166107
test loss =8.584201249628341
real
pred
Close
Date
0
10
20
30
40
50
60
70
80
2006
2008
2010
2012
2014
2016


crvl_data_7
train loss =0.3108042836110641
test loss =0.8689647485305956
real
pred
Close
Date
10
20
30
40
50
60
2006
2008
2010
2012
2014
2016



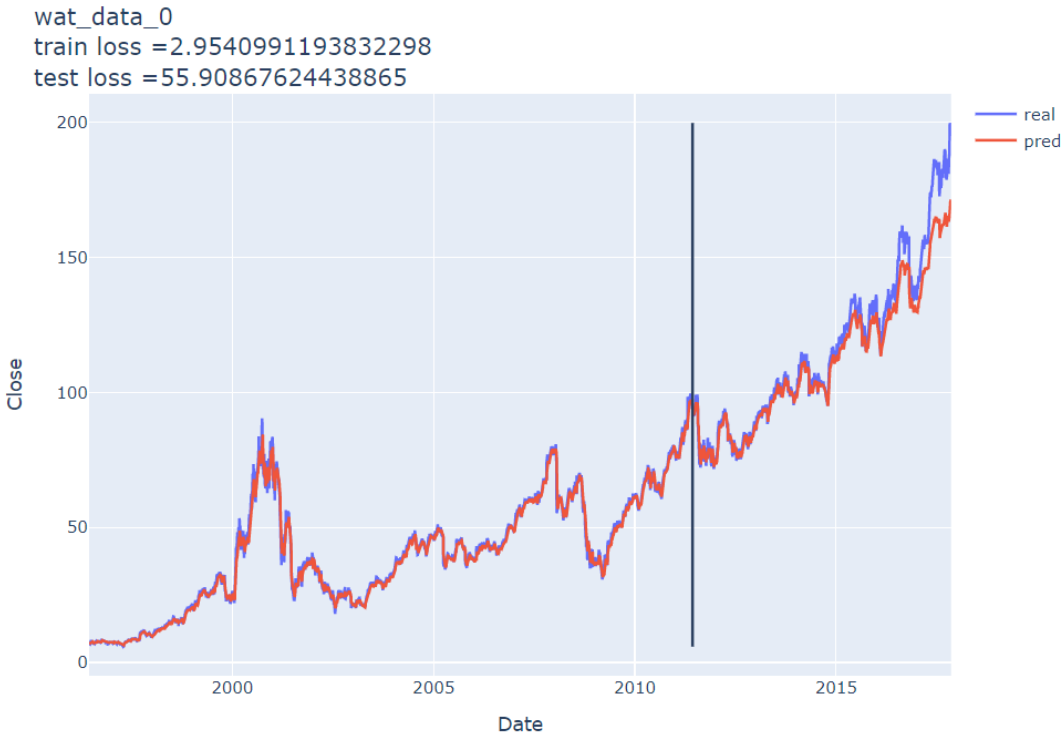


从上面十张图可以看出：当利用简单的 LSTM 结构，与较为简单的训练集训练模型时，该模型在收盘价标准差不大的公司（排名 2 之后）上表现良好，这也说明随着收盘价标准差的增大，数据的预测变得更有挑战性，我们应该引入更为复杂的训练集或更为复杂的 LSTM 结构：

- 复杂的训练集意味着我们增加更多更好的新特征并且扩充训练数据的量；
- 复杂的 LSTM 结构启发我们进行参数的调整，增加验证集并利用网格搜索法（Grid Search）选出最佳的一组参数进行预测。

接下来我们将展示在标准差最大的公司上，添加不同的特征组合后的训练集训练得到的模型在测试集上的效果：

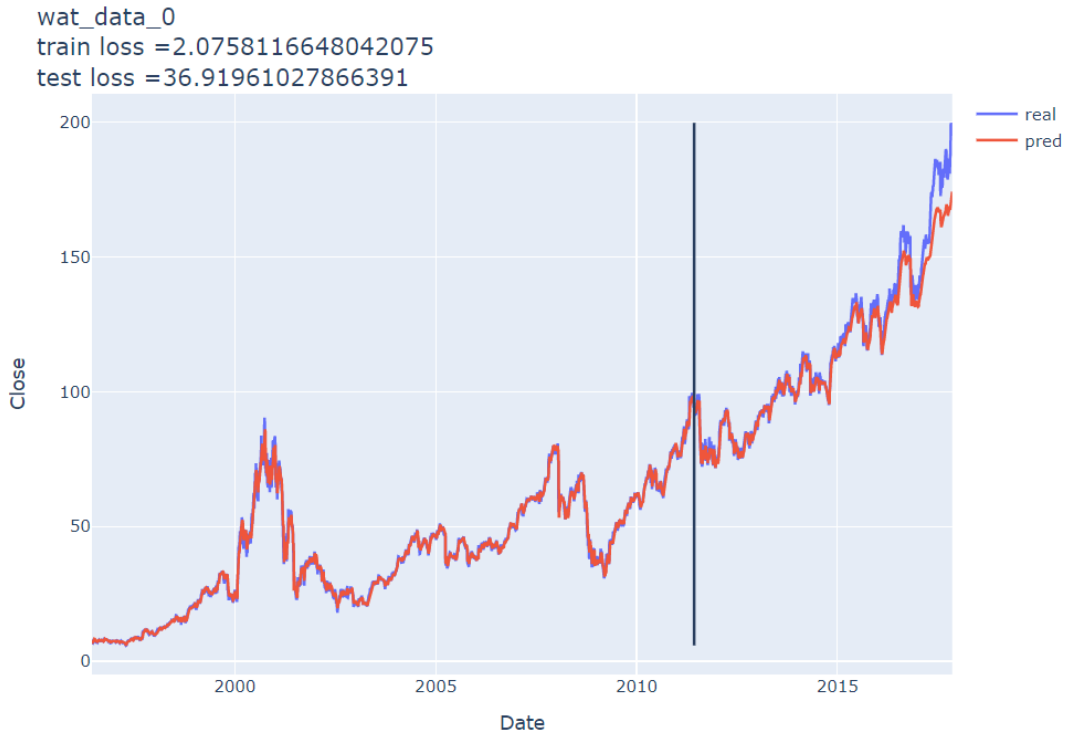
2. 添加历史标准差、均值



3. 添加历史最大值、最小值

4. 添加三个技术指标（RSI、MACD、OBV）

5. 添加历史标准差、均值、最大值、最小值



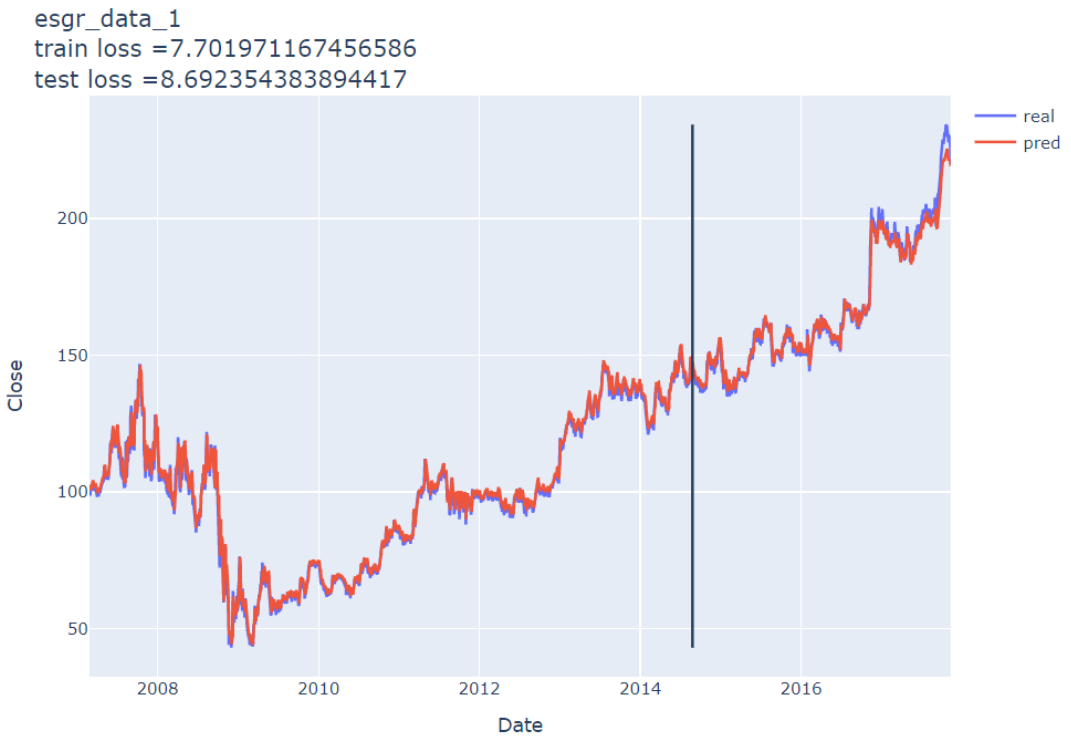
6. 添加历史标准差、均值和三个技术指标

由上面几组训练集、测试集的损失值的对比可知，对于该公司来说，利用增加特征：历史标准差、均值、最大值和最小值的训练集，所得到的模型的泛化能力最强。而相比之下增加三个技术指标甚至会降低其在测试集上的预测能力。这

一结果也说明：并不是特征越多越复杂，最终得到的模型表现越好。这也是特征工程的意义所在：找到最适合研究对象的特征组合，从而提高模型的预测能力。

7. 扩大训练集样本数目

增加新特征固然可以在一定程度上提高模型的泛化能力，而扩大训练集样本数也能够达到这样的目的，在进行模型训练，尤其是神经网络，的时候，充足的训练样本是保证模型最终效果的重要因素。同时，当我们增加训练数据量的时候，训练模型所消耗的时间也相应增加，此时使用批训练可以有效提高训练模型的效率，将耗时缩短至原本的 1/5。下面展示在添加了近乎 9 倍的训练样本数且不添加新特征后，模型在各个公司的数据上的表现情况：



对比之前的预测效果，可以看出，使用样本数更大的数据训练出的模型的泛化能力有了明显的提升，这也符合我们对于欠拟合的处理方式的认知。

8. 实验结果总结展示

下面将实验的所有结果以表格的形式进行展示，展示内容为：添加不同特征

组合的训练集训练出的模型在各自公司上的表现情况（以训练集与测试集上的损失值大小体现）与使用样本数更多的训练集训练出的模型的表现情况。

表格 1 训练集损失

	Orig	Stat1	Stat2	Fina	Stat12	Stat1Fina	TenComp
0	2.173	2.954	1.892	2.377	2.076	2.532	2.277
1	19.072	7.275	6.389	6.420	9.276	7.318	7.702
2	0.491	3.994	0.850	0.520	0.514	0.635	0.591
3	2.518	3.110	2.968	2.790	2.952	2.683	1.715
4	3.247	2.572	2.733	2.343	2.653	2.354	2.392
5	0.604	0.566	0.502	0.589	0.649	0.611	0.465
6	0.763	0.544	0.933	0.700	0.701	0.792	0.498
7	0.311	0.298	0.455	0.317	0.450	1.426	0.368
8	0.529	0.481	0.428	0.421	0.381	0.428	0.604
9	1.692	1.515	1.595	1.561	1.528	1.467	1.941

表格 2 测试集损失

	Orig	Stat1	Stat2	Fina	Stat12	Stat1Fina	TenComp
0	57.363	55.909	85.123	162.366	36.920	235.379	6.740
1	25.641	51.115	14.936	38.525	56.058	99.268	8.692
2	606.111	106.183	294.632	348.024	302.512	660.400	1.640
3	2.593	3.417	2.956	2.926	3.699	3.446	1.675
4	5.776	4.744	4.416	4.980	4.545	6.330	4.518
5	2.661	4.070	3.370	3.584	9.613	4.084	1.475
6	8.584	3.126	4.551	6.973	2.561	19.952	1.455
7	0.869	0.850	1.382	1.060	3.028	6.734	1.103
8	2.328	1.275	1.473	1.767	1.306	1.441	1.362
9	1.026	1.007	0.995	0.944	1.005	1.145	1.289

以上两个表格分别展示了十家公司的训练集进行不同处理后训练出的模型在本公司训练集和测试集上的损失函数值，其中：第一列为各家公司的日收盘价

排名，Orig、Stat1、Stat2、Fina、Stat12、Stat1Fina 和 TenComp 分别代表不添加新特征、增加历史标准差和均值、增加历史最大值和最小值、增加三个技术指标（RSI、MACD 和 OBV）、增加历史标准差均值最大值和最小值、增加标准差均值和三个技术指标以及利用其它九家公司的部分数据来扩大训练集样本数（此时并没有添加任何新特征）。

通过上面两个表格内容的对比可以直观地感受到，增加合适的新特征在一定程度上能够提高模型的泛化能力，而提升的效果也与所处理的数据的挑战性（标准差大小）有关；对于日收盘价标准差较大的公司，数据的分析预测工作更有挑战性，而增加训练集样本数可以显著地提升模型的预测效果，这也证明了神经网络需要大量的数据才能保证其良好的输出。

(八)实验总结

本次实验利用 LSTM 实现对股市日收盘价的预测，期间利用滑动窗口的方式构造三维输入项，并且在训练模型时，尝试添加不同的特征，比较统计学特征与金融学特征在各个公司上的表现情况。同时，通过增加训练集样本数使得模型得以充分训练，具有较强的泛化能力。在保证模型预测效果的前提下，使用批训练的方式，能够有效地减少训练模型消耗的时间。

(九)实际意义说明

在本次实验中，不论是训练还是测试时，我们使用到的特征全部都是真实值，这也就涉及到本次实验的实际意义：

我们本次所做到的只是利用当天（例如第 n 天）之前的信息，来预测当天（第 n 天）的收盘价，而不是后面所有天（第 n 天至结束）的收盘价，即我们所做的工作仅是单步预测。第 n 天结束，我们将获得第 n 天的信息，第 n+1 天的预测结果是基于第 n 天结束时获取到的信息以及之前的信息，通过训练好的模型得到。这里可以区分 ARIMA 对于后面所有天的预测，即多步预测，ARIMA 是利用前 n 天的时间序列数据确定参数 p、d、q，对于 n+i 天的预测是基于前 n 天的真实值和第 n+1 到第 n+i-1 天的预测值。这也是为什么模型在后期也可以表现得很好，因为我们使用的信息永远都是真实的，而不是基于上一次的预测结果的迭代。

总结与展望

(一)总结

时间序列与我们的日常生活息息相关，而近年来，随着机器学习和深度学习的进一步发展，利用非传统方法分析时间序列为广大数据分析者提供了新的思路。对于股票这一时间序列的预测一直是金融市场中的热门研究领域，本文主要研究利用循环神经网络的变体 LSTM 进行股票市场趋势的预测。

由于原始数据信息提供的特征较少，本文利用特征工程为数据添加了大量新特征，提高信息利用率，并通过增加训练集样本数的方式，使得神经网络模型能够被充分训练，大幅度提高模型的泛化能力。主要工作有如下几点：

1. 在进行特征工程时，添加统计学与金融学上的新特征：历史标准差、均值、最大值、最小值、RSI、MACD 和 OBV 三个指标，并对比不同特征组合对于不同公司训练集训练得出的模型的泛化能力进行对比；
2. 考虑到神经网络需要大量的训练样本才能保证其预测能力，后期，我们对于单一公司的 LSTM 模型训练时，利用了其他九家公司的数据信息，即添加了大量的训练集样本，实验结果证明此方式显著地提升了模型的预测效果；
3. 神经网络的训练往往需要耗费大量的时间，为了在保证模型泛化能力的同时尽可能减少训练耗时，本文设计了简单易操作的批训练方式，使得训练耗时缩短至原来的五分之一；
4. 同时，对于最终预测结果损失函数值为何如此小，本文第三章最后也给出了本次实验的实际意义的解释：每次的预测都是建立在该天之前的信息（即输入模型的 X 值）完全真实的情况，是单步预测，并与我们熟知的 ARIMA 多步预测进行这个意义上的对比。

(二)未来工作与展望

本文利用深度学习的 LSTM 模型对于股票市场的日收盘价进行了预测，并取得了较好的效果，但本文的研究仍有不足，需要未来进行进一步的研究与探索，主要表现为以下几个方面：

1. 本文使用 LSTM 进行预测本质上是一种单步预测，但这明显很难满足股

票市场参与者的需要，未来可以进一步改进模型，使其具有多步预测的能力；

2. 本文的研究主要围绕着添加不同特征组合与增加训练集样本数对于提升模型泛化能力的影响，对于模型本身的参数设置并没有过多的研究，未来可以添加验证集并使用一些调参技巧，如网格搜索或贝叶斯优化等，对于模型参数进行调整，进一步提高模型的泛化能力；
3. 本文在进行特征缩放时，出现了一种罕见的现象：通常进行多特征回归时，若特征量纲不同，对于特征需要进行缩放。如果回归的目标（y）是一维的，则不需要进行缩放，如果是多维，且量纲不同，则同特征情况类似，需要进行缩放处理，否则会影响到模型的拟合效果。但是，在本次实验中，即使回归的目标，日收盘价，是一维的，若不进行缩放处理，最终预测的结果会有一个阈值，不论实际值超出该阈值多少，预测值始终在阈值之下。针对该问题，查阅了多个文献论坛，也没能找到合理的解释，这一点是值得未来进行研究探索的。

参考文献

[1]张旭. 基于循环神经网络的时间序列预测方法研究[D].南京大学,2019.

[2]张金磊,罗玉玲,付强. 基于门控循环单元神经网络的金融时间序列预测[J]. 广西师范大学学报(自然科学版),2019,37(02):82-89.

[3]杨丽,吴雨茜,王俊丽,刘义理. 循环神经网络研究综述[J]. 计算机应用,2018,38(S2):1-6+26.

[4]张蜀林,赵雄飞. LSTM 模型在中国 A 股市场的应用[J]. 全国流通经济,2018(35):94-95.

[5]贾宇杰. MACD 指标在股市实战中的应用及其存在的问题研究[J]. 中国商论,2018(12):34-36.

[6]龙奥明,毕秀春,张曙光. 基于 LSTM 神经网络的黑色金属期货套利策略模型[J]. 中国科学技术大学学报,2018,48(02):125-132.

[7]李泽龙,杨春节,刘文辉,周恒,李宇轩. 基于 LSTM-RNN 模型的铁水硅含量预测[J]. 化工学报,2018,69(03):992-997.

[8]毛景慧. 基于 LSTM 深度神经网络的股市时间序列预测精度的影响因素研究[D].暨南大学,2017.

[9]王征韬. 深度神经网络压缩与优化研究[D].电子科技大学,2017.

[10]杨祎玥,伏潜,万定生. 基于深度循环神经网络的时间序列预测模型[J]. 计算机技术与发展,2017,27(03):35-38+43.

[11]孙瑞奇. 基于 LSTM 神经网络的美股股指价格趋势预测模型的研究[D].首都经济贸易大学,2016.

[12]曾勇. MACD 指标基本原理及其运用[J]. 商,2013(20):336.

[13]M. Shanker,M.Y. Hu,M.S. Hung. Effect of data standardization on neural network training[J]. Omega,1996,24(4).

[14]闫洪举.基于深度学习的金融时间序列数据集成预测[J].统计与信息论坛,2020,35(04):33-41.

[15]Songqiao Qi,Kaijun Jin,Baisong Li,Yufeng Qian. The exploration of internet finance by using neural network[J]. Elsevier B.V.,2020,369.

[16]杨青,王晨蔚.基于深度学习 LSTM 神经网络的全球股票指数预测研究[J].统计

研究,2019,36(03):65-77.

[17]邓凤欣,王洪良.LSTM 神经网络在股票价格趋势预测中的应用——基于美港股票市场个股数据的研究[J].金融经济,2018(14):96-98.

[18]王渊明. 基于 LSTM 神经网络的电商需求预测的研究[D].山东大学,2018.

[19]黄婷婷. 机器学习在金融领域的应用[D].中国科学技术大学,2018.

[20]李伟.基于 LSTM 模型的迁移学习预测外汇汇率[J].消费导刊,2019,(5):186-187. DOI:10.3969/j.issn.1672-5719.2019.05.146.

[21] S. Majorana and L. Chua, "A Unified Framework for Multilayer High Order CNN", Int'l Journal of Circuit Theory and Applications, 26:567-592, 1998.

[22] 韩沙.基于.NET 平台的股票分析系统的设计与实现[D].四川:电子科技大学,2012. DOI:10.7666/d.D771212.

致谢

毕业设计从选题到实验再到最后论文编写一共用了快 4 个月时间，期间有过对研究意义的迷茫，也有代码写不出来时的沮丧，更有过对自我的怀疑，但是最终，这些难关都在老师和同学们的帮助下顺利度过。

首先，非常庆幸能够选择范红军老师作为论文的指导老师，也很感谢老师选择了我。范老师工作负责，在选题初期就与我进行多次交流，根据我的意愿为我提供合适的选题。前期为我提供很多有价值的阅读材料，帮助我系统化理解研究主题。在整个实验和论文编写过程中，范老师对于我提出的疑问都及时进行了反馈和建议，帮助我顺利完成一篇内容充实、条理清晰的论文。

其次，很感谢我实习的老板 Phil Cui，虽然因为疫情，没能去公司实习，但是 Phil 依旧耐心地对我进行指导，与其说他是工作的老板，不如说是我的第二位导师。初期的我对于神经网络只有很浅层的理解，在实验过程中遇到很多的难题，单凭我当时所学并没有办法解决，是 Phil 建议每周远程会议一次，针对我遇到的困难，提供专业上的指导，也正是因为他提供的建议，我才能找出代码中的问题并逐步改进模型。更重要的是，因为初期实验不理想，我有过自我怀疑的阶段，是 Phil 告诉我保持本心，循序渐进才是做研究的态度，将我从浮躁的状态拉出来。未来我也将保持这种态度进行学习研究。

然后，我要谢谢我的父母，无论什么时候，父母永远是我的依靠。在生活上，他们一直尽力为我提供最好的后勤保障工作；在学习上，不论是出国交流还是在国内升学，他们总是支持我的选择，给予我最大的信任。在我取得成绩时，他们会在我身旁为我庆祝；在我遇到瓶颈时，他们还是会在我身后给我宽慰。以后我也会继续认真学习，不辜负父母的期望。

最后，想感谢在这珍贵的四年里我所遇到的每一个人，大学生活给予我的不仅是丰厚的知识，更是在日常中所培养出的思维方式和为人处事的方法。无论是在学习、生活还是在工作上，是你们的关怀和鼓励让我度过了充实的四年。

毕业论文完成之际，我的心情依旧无法平静，大学生活的种种在我的脑海里浮现，在此，我再次真挚地向这四年内帮助过我的你们表示感谢！

附录

本文实验使用 Python 编程语言实践，附录为代码和所有实验结果的 GitHub 地址：

https://github.com/lethe-ye/stock-prediction/blob/master/OneComp.py

https://github.com/lethe-ye/stock-prediction/blob/master/TenComp.py

https://github.com/lethe-ye/stock-prediction/blob/master/figures.7z