武汉大学 经济与管理学院
Economics and Management School of Wuhan University

时间序列方法+机器学习在股指预测中的应用

—— 基于 LSTM-RF Regression 方法

刘郅哲

经济与管理学院

武汉大学

2022 年 *5* 月 *25* 日

1 研究背景及目的

1.1 股票指数预测

股指预测一直是全金融市场关注的重点问题，在过去的方法上可分为线性预测模型和非线性预测模型。

线性预测模型包括移动平均自回归模型（ARIMA）、广义自回归条件异方差模型（GARCH）、指数平滑预测模型等，上述传统时间序列模型的应用使得金融市场预测研究取得了很大进步（刘国旗，2000[1]；惠晓峰等，2003[2]）。

鉴于金融时间序列的随机游走特性及低信噪比特征，实现精准预测仍然非常困难，加之自变量和因变量之间的关系通常会随时间发生动态变化，致使传统时间序列模型很难有效进行金融市场预测。

1.2 深度神经网络在股指预测的过往研究

近年来，深度学习方法开始出现在金融预测中，Xiong 等（2015[3]）利用 LSTM 神经网络对标普 500 指数的波动率进行建模，结果表明 LSTM 神经网络对包含噪声的金融时间序列数据具有预测潜力；Di Persio 和 Honchar（2016[4]）使用 MLP、CNN 和 LSTM 神经网络对标普 500 指数第二天收盘价涨跌进行预测，发现基于 CNN 的预测误差最小；Di Persio 和 Honchar（2017[5]）将 RNN、LSTM 神经网络和 GRU 神经网络用于谷歌股价趋势预测，结果显示 LSTM 神经网络在金融序列预测方面具有优势。

1.3 时间序列方法与机器学习的结合

在 2018 年大数据与人工智能国际会议（ICBDAI 2018）上，Mei 等[6]提出了基于 ARIMA 和 SVR 相结合的预测方法，其中，SVR 模型负责对 ARIMA 的预测残差进行拟合，得到了较好的预测效果。

1.4 研究目的

本文作为 Project2 的优化探索，借鉴 Mei 等的时间序列方法与机器学习方法的研究思路，意在利用 LSTM 方法并利用 Random Forest Regression 对残差进行拟合从而对市场指数进行更加精确的预测。其经济理论基础为利用

LSTM 对趋势及价格时间序列带来的市场信息进行解释，然后采用 RF 对其他市场信息（如情绪）进行解释，二者组合能够较全面地解释市场信息。通过更加精确的时间序列预测，做出指数基金产品的择时交易策略或通过休哈特控制图来实现异常值的检测实现风险控制。

2 数据收集与预处理

2.1 数据来源

所选股指为沪深 300 指数（000300.SH），数据采集于 Wind 数据库与 Choice 数据库，获取了包括开、收、高、低等常见量价指标与 MACD、ADTM 等衍生技术指标、市场情绪指标的 16 个特征，其总览如下表：

表 1：变量表

特征	变量名称
开盘价	*open*
收盘价	*close*
日内最高价	*high*
日内最低价	*low*
当日涨跌幅	*ret*
换手率	*turnrate*
成交量	*volume*
动态买卖气指标	*ADTM*
均幅指标	*ATR*
顺势指标	*CCI*
异同移动平均	*MACD*
动量指标	*MTM*
变动率指标	*ROC*
盈潮-S 指标	*SOBV*
标准差（26 周期）	*STD_26*
标准差（5 周期）	*STD_5*

对该 16 个变量取 2010-01-04 至 2022-5-17 共 3003 个交易日的数据进行数据的预处理。

2.2 数据预处理

2.2.1 归一化

利用 sklearn 库中 MinMaxScaler 对数据进行归一化处理，处理过程如下：

$$X_{std} = \frac{X - X.min}{X.max - X.min}$$

$$X_{scaled} = X_{std}(rangemax - rangemin) + rangemin$$

此处，将全部数据通过归一化映射进$(-1, 1)$的区间内，方便后续处理。

2.2.2 张量化

为将数据传进神经网络，采用 torch.FloatTensor()函数将价格序列提取数值后转换成[3003,1]的张量。

2.2.3 窗口化

LSTM 网络支持滑动窗口预测，故设定输入窗口为 30，输出窗口为 1。

2.2.4 批次化

LSTM 网络层仅支持定义批量的批次数据传入，故设定每批次传入数据长度为 30，同输入窗口长度，并将序列张量标记，标记在新的维度。同时对不足长的批次进行补长。

2.2.5 训练集测试集划分

对长序列进行 7：3 的划分，进行必要的基于滑动窗口长度的裁剪后，得到训练集为[2371, 2, 30]的张量，测试集为[570, 2, 30]的张量，再将训练集进行 7：3 的划分，分出实际训练集与验证集。

3 数据描述性统计

在对数据进行归一化处理后，数据量纲消失，能够得到更好的统计性质。对特征数据进行描述性统计，可以得到全部数据的中位数、均值相距较近，从总体上看，指数的价格在近十年内存在上涨趋势，但近期跌幅较大。

详细描述统计信息见下表：

表 2：变量描述性统计表

Variables	Min	Max	Mean	Median
open	*-0.2386*	*1.0000*	*0.3540*	*0.4282*
close	*-0.2242*	*1.0000*	*0.3957*	*0.4708*
high	*-0.2303*	*1.0000*	*0.3631*	*0.4351*
low	*-0.2053*	*1.0000*	*0.4072*	*0.4851*
ret	*-0.8879*	*0.8646*	*0.1318*	*0.1380*
volume	*-0.7785*	*0.1854*	*-0.5455*	*-0.5771*
turnrate	*-0.9049*	*-0.0976*	*-0.7101*	*-0.7360*
ADTM	*-0.9768*	*0.9559*	*0.2235*	*0.2848*
ATR	*-0.9517*	*0.5744*	*-0.6919*	*-0.7400*
CCI	*-1.0000*	*0.8080*	*0.0725*	*0.0916*
MACD	*-0.4227*	*0.9582*	*0.1225*	*0.1163*
MTM	*-0.3076*	*0.9834*	*0.2463*	*0.2534*
ROC	*-0.6364*	*0.6266*	*-0.0410*	*-0.04074*
SOBV	*0.6398*	*1.0000*	*0.8497*	*0.8391*
STD_26	*-0.85681*	*-0.0237*	*-0.5389*	*-0.6154*
STD_5	*-0.9627*	*0.3265*	*-0.6695*	*-0.7192*

4 模型构建

4.1 长短期神经网络（LSTM）

4.1.1 模型介绍（RNN-LSTM）

循环神经网络（Recurrent Neural Network, RNN）是一种用于处理序列数据的神经网络，其在处理自然语言模型、时间序列上优于传统的多层感知机（MLP）。RNN 相较于一般多层感知机，添加了计算含有过去时间状态（隐状态）的隐藏层，该层由于需要循环计算，被称为循环层（Recurrent Layer）。

普通 RNN 的结构如下：

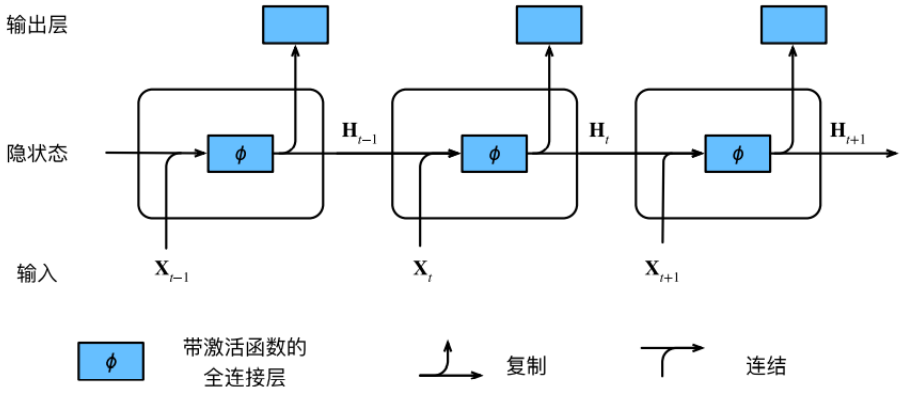

图 1：RNN 网络结构

长短期记忆神经网络（Long Short-Term Memory, LSTM）是一种特殊的 RNN，为解决长序列训练过程中的梯度消失、梯度爆炸问题而产生，在处理长序列时较 RNN 表现更好。

LSTM 引入了记忆元，用于记录附加的信息（类隐状态），其门控结构基本为输出门：从单元中输出，输入门：何时将数据读入单元，遗忘门：重置单元内容。又引入了候选记忆元以控制遗忘和输入，输入门控制采用多少来自候选记忆元的新数据，遗忘门控制保留多少过去的记忆元的内容，其神经元结构如下图：

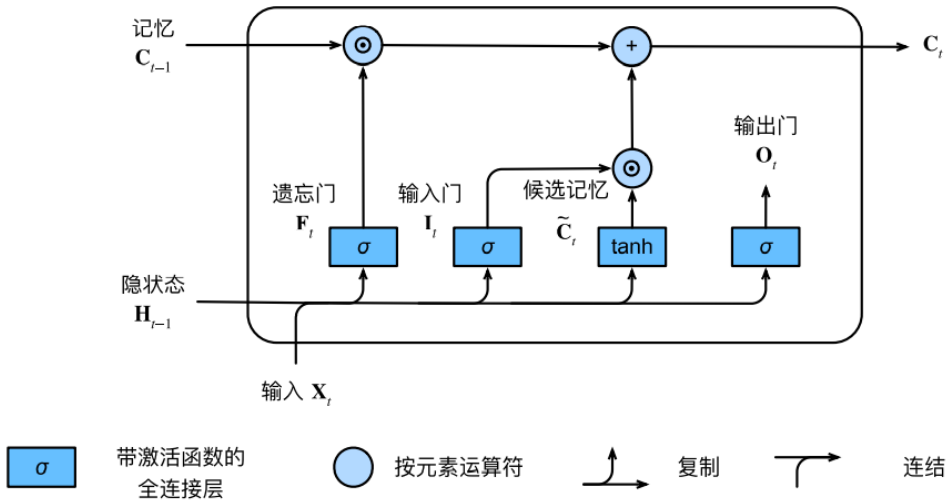

图 2：LSTM 的神经元结构

简言之，即相较于 Naïve RNN，LSTM 结构除对隐状态进行处理之外，也同时对过去记忆进行处理并结合加入输出，主要特点为记忆性。

4.1.2 网络模型搭建

在 pytorch 库中 nn.LSTM 类继承了 nn.rnn 类，其定义相仿。预先设置的隐藏层维数为 100，输入长度、输出长度与设置窗口长度相同，分别为 30 与 1。将网络模型移至 GPU 训练以提高训练效率。

```
class LSTM(nn.Module):
    def __init__(self, input_dim = 30, hidden_layer_dim = 100, output_dim = 1):
        super().__init__()

        self.hidden_layer_dim = hidden_layer_dim

        self.lstm = nn.LSTM(input_dim, hidden_layer_dim).cuda()

        self.linear = nn.Linear(hidden_layer_dim, output_dim).cuda()

        self.hidden_cell = self.init_hidden()

    def init_hidden(self):
        return(torch.zeros(1, 1, self.hidden_layer_dim).cuda(),
               torch.zeros(1, 1, self.hidden_layer_dim).cuda())

    def forward(self, input_seq):
        lstm_out, self.hidden_cell = self.lstm(input_seq.view(len(input_seq), 1, -1),
                                               self.hidden_cell)

        predictions = self.linear(lstm_out.view(len(input_seq), -1))

        return predictions[-1]
```

4.1.3 训练

4.1.3.1 梯度裁剪

采用裁剪方法，防止在反向传播过程中，随层数增加而产生的梯度爆炸问题，将超出阈值的梯度值锁定在阈值，防止跳过最优解。

4.1.3.2 动态学习率

在学习过程中，添加学习率递减的模块，随训练步数的增加不断降低学习率，令训练初期模型收敛较快，后期趋于稳定。

4.1.3.3 超参定义

该模型训练过程需要定义的超参为初始学习率 lr，学习率衰减率 gamma，总步数 epochs。

经过在验证集上的人工调整，最终得到的较优参数组合为 lr = 0.02，gamma = 0.96，epochs = 150。

```
def train(train_data):
    model.train()
    for batch_index, i in enumerate(range(0, len(train_data) - 1, batch_size)):

        #按批获取数据、标签
        batch_data, targets = get_batch(train_data, i, batch_size)
        #总损失
        total_loss = 0
        #隐藏层、梯度清零
        model.hidden_cell = model.init_hidden()
        optimizer.zero_grad()
        #前馈并计算损失
        output = model(batch_data)
        loss = criterion(output, targets)
        #反向传播
        loss.backward()
        #梯度裁剪
        torch.nn.utils.clip_grad_norm_(model.parameters(), 0.7)
        optimizer.step()
        #打印训练步骤
        total_loss += loss.item()
        log_interval = int(len(train_data) / batch_size / 5)
        if batch_index % log_interval == 0 and batch_index > 0:
            cur_loss = total_loss / log_interval
            print('| epoch {:3d} | {:5d}/{:5d} batches | lr {:02.6f} | loss {:5.5f} | ppl {:8.2f}'
                .format(epoch, batch_index, len(train_data) // batch_size, scheduler.get_lr()[0],
                cur_loss, math.exp(cur_loss)))
```

4.1.4 模型结果

经过训练后，模型的预测结果产生了 LSTM 常见的问题：预测延后、不够精确。单一 LSTM 网络不能精确预测股指序列，我们引入随机森林对残差进行进一步学习的初始思路是正确的。

图 3：LSTM 预测结果（红色为预测值）

4.2 随机森林回归（Random Forest Regression）

4.2.1 模型介绍

随机森林（Random Forest）是一种将 bootstrapping 与 bagging 结合起来的集成树学习模型。随机森林构造了多个决策树，其特点为随机取样本、随机取特征，所以每棵树存在相关性的同时也存在异质性，当需要对某个样本进行预测时，统计森林中的每棵树对该样本的预测结果，然后通过投票法从这些预测结果中选出最后的结果。随机森林回归（Random Forest Regression）即为构造多颗回归树，由投票加权算法得出最终预测值。

4.2.2 模型搭建、训练

通过对表 1 中特征对 LSTM 结果产生的残差进行回归预测，训练随机森林模型。定义随机森林的超参数（初始定义出于效率与泛化能力的考虑）为随机森林中树的数目 ntree = 1000， 终端结点的最小大小 nodesize = 25，随机抽样候选变量数 mtry = 3。该模型的全数据集为 LSTM 的测试集，划分训练集共 400 个样本，测试集共 168 个样本。训练和预测过程采用滚动建模，即每预测一期后将该期特征值与实际值纳入训练集重新训练、预测。

```
forestmodel <- randomForest(diff ~ ret + turnrate + ADTM + ATR + CCI + MACD + MTM + ROC
+ SOBV+ STD_26 + STD_5,
                        data_train,
                        ntree = 1000,
                        mtry = 3,
                        nodesize = 25)
for(i in 1:168){
  predictForest[i] <- predict(forestmodel, type = "response", newdata = data_test[i,])
  data_train <- bind_rows(data_train, data_test[i,])
  forestmodel <- randomForest(diff ~ ret + turnrate + ADTM + ATR + CCI + MACD + MTM +
ROC + SOBV+ STD_26 + STD_5,
                        data_train,
                        ntree = 1000,
                        mtry = 3,
                        nodesize = 25)
}
```

4.2.3 模型结果

采用随机森林回归对股指序列进行独立拟合，效果较差，如下图：

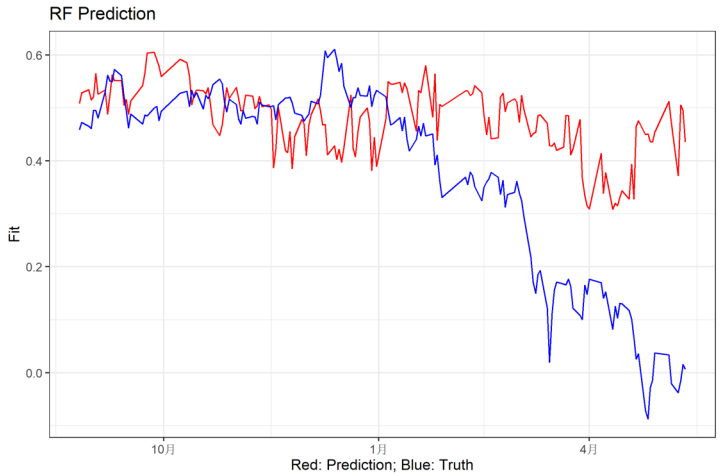

图 4：随机森林回归模型预测

对残差序列进行拟合，并将随机森林回归的拟合值与 LSTM 模型的预测值加总后，所得序列图如下：

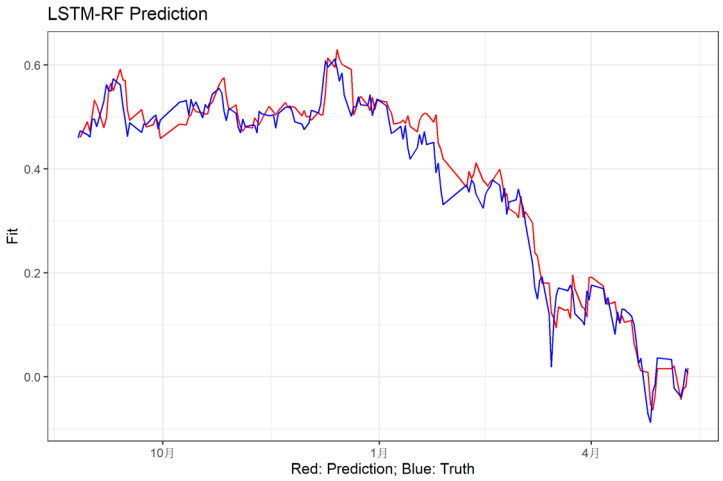

图 5：LSTM-RF 模型预测

三种预测模型的均方根误差如下：

表 3：不同模型的均方根误差表

Model	$RMSE = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n}}$
LSTM	0.08149
RF	0.40689
LSTM-RF	0.03592

可见，在均方根误差的衡量角度上，LSTM-RF 模型预测表现最好，精度最高，较单一 LSTM 模型减少了 50%左右的预测误差。

5 研究结论

• 时间序列方法预测趋势+机器学习拟合特征的思路是正确的，模型效果也是有效的；

• 尽可能克服了 LSTM 的时滞弱点，在随机森林回归模型的加入下，整体鲁棒性较高；

• 较为精确的预测能够给指数产品的量化投资提供方向指引与研判；

• 在风险管理领域，能够识别较大程度的跌幅，有利于机构与个人避险。

6 优化展望

• 参数调优仍然需要探索，LSTM 模型细节仍待优化，考虑后续引入同源的 GRU 模型或引入注意力机制神经网络架构（Transformer）进行预测对比；

• 鉴于神经网络的高度可调节性，可以添加多种非同质信息作为神经网络的输入，附加小波分解或主成分分析等数据预处理技术进行模型优化。

---

[1] 刘国旗. 非线性 GARCH 模型在中国股市活动预测中的应用研究 [J]. 统计研究, 2000(1): 49-52.

[2] 惠晓峰, 等. 基于时间序列 GARCH 模型的人民币汇率预测 [J]. 金融研究, 2003(5): 99-105.

[3] Xiong R, Nichols E P, Shen Y. Deep learning stock volatility with google domestic trends [J]. arXiv preprint arXiv: 1512.04916, 2015.

[4] Di Persio L, Honchar O. Artificial Neural Networks Architectures for Stock Price Prediction: Comparisons and Applications: Comparisons and Applications [J]. International Journal of Circuits, Systems and Signal Processing, 2016(10): 403-413.

[5] Di Persio L, Honchar O. Recurrent Neural Networks Approach to the Financial Forecast of Google Assets [J]. International Journal of Mathematics and Computers in Simulation, 2017(11): 7-13.

[6] Wenjuan Mei, Pan Xu, Ruochen Liu, and Jun Liu. Stock price prediction based on ARIMA - SVM model [C]. 2018 International Conference on Big Data and Artificial Intelligence(ICBDAI 2018)