一、模型选择

何为深度神经网络?顾名思义,深度神经网络与更常见的单一隐藏层神经网络的区别在于深度,即数据在模式识别的多步流程中所经过的节点层数。传统机器学习系统主要使用由一个输入层和一个输出层组成的浅层网络,至多在两层之间添加一个隐藏层。三层以上(包括输入和输出层在内)的系统就可以称为“深度”学习。所以,深度是一个有严格定义的术语,表示一个以上的隐藏层。在深度神经网络中,每一个节点层在前一层输出的基础上学习识别一组特定的特征。随着神经网络深度增加,节点所能识别的特征也就越来越复杂,因为每一层会整合并重组前一层的特征。

为何选择 LSTM? 全连接的深度神经网络存在着一个问题——无法对时间序列上的变化进行建模。然而,样本出现的时间顺序对于自然语言处理、语音识别、时间序列分析等应用非常重要。为了适应这种需求,就出现了另一种神经网络结构——循环神经网络 RNN。在 RNN 中,神经元的输出可以在下一个时间戳直接作用到自身,即第 i 层神经元在 m 时刻的输入,除了(i−1)层神经元在该时刻的输出外,还包括其自身在(m−1)时刻的输出。而 RNN 仍存在一个显著问题，因为神经元数目和结构受限，只能利用前面几个时间点的数据对当前时间点的数据进行预测，而较长时间之前的数据难以被应用进来。但在股指预测中，因为“历史会不断重演”的特性，长久之前的数据也可能对当前时间的预测有较大的影响，因此 RNN 的这个问题更为显著，需要对 RNN 的结构进行改进，也就是我们所用的 LSTM 网络。

LSTM 通过修改 RNN 神经元的结构，添加控制输入的三个“门”，有了明显的“选择性记忆”特性。能够记忆对当前影响较大的数据，而遗忘对当前影响较小的数据，很好的克服了 RNN 记忆受限的问题。因此理论上非常适合用 LSTM 来进行预测。

二、数据处理

对于机器学习来说，数据处理与特征选择至关重要。对于给定的数据集，如果其中存在不合理的畸形数据，在训练时会产生非常严重的干扰，极大影响学习的性能和预测的准确率。因此，首先要做的就是对数据集进行筛选和清理。步骤如下：

（一）提取主力合约

考虑到不同年份之间的主力合约数据可能会有较大差距，仅提取 2016 年的主力合约数据进行训练和测试。提取每天的 IF 主力合约，拼接为最终的主力合约。

（二）数据筛选与清理

1.删除非正常交易时间的数据，交易时间为上午 9：30-11：30，下午 1:00 到 3:00
2. 因为每天开盘的前 10 分钟和收盘的最后 10 分钟数据波动较大，暂时在训练时去掉这些数据
3. 一分钟之内，最大股价和最低股价相差超过 50 点的，那么应该尽量删掉这些数据
4. 删除一分钟成交量大于 1000 的，这些数据可能会影响整体数据

（三）特征选择

每个分钟数据都有诸多特征，其中的一些特征还被经常用作交易策略的编写中。有必要把这些特征统统加进去。本模型加入的特征有：

1.主力合约中原有的特征：

“Latestprice”，最新价，一分钟内最后一个价格
“First_Latestprice”，首个最新价，一分钟内第一个出现的价格
“MaxPrice”，最高价，一分钟内的最高价
“MinPrice”，最低价，一分钟内的最低价
“Last_Buy1price”，最新买一价，一分钟内最后一个买一价
“Last_Sell1price”，最新卖一价，一分钟内最后一个卖一价
“Last_Buy1quantity”，最新买一量，一分钟内第一个买一量
“Last_Sell1quantity”，最新卖一量，一分钟内第一个卖一量

“Stockup”，增仓，一分钟内增仓量的和

“Turnover”，成交额，一分钟内成交额的总和

“Volume”，成交量，一分钟内成交量的总和

2.MA、EMA、MACD 等指标：

分别加入收盘价和成交量的 5 分钟、12 分钟、26 分钟的 MA（移动平均线）、EMA（指数移动平均线）指标，以及几条移动平均线之间的距离指标。共（3+1）*2*2 = 16 个指标。

计算 EMA_12 与 EMA_26 的离差 DIFF 指标，离差平均值 DEA 指标，进而得到 MACD（平滑异同平均线）指标。共 3 个指标。

3.RSI、布林带等指标：

该部分指标在 V1.0 版本中暂时未加入，等到后续进行模型优化时再陆续加入。

（四）数据归一化

归一化是使不同的量纲之间具有可比性,比如属性 A 的值的范围为 2000 左右,属性 B 的值范围 100 左右,为了使其两者之间具有可比性,分别对属性 A 和属性 B 进行归一化处理,本实验采用线性归一化方式处理数据,其结果范围在 0-1 之间,公式如下:

$$x_t^i := \frac{x_t^i - \min x^i}{\max x^i - \min x^i}$$

（五）组时间序列

我们要研究的问题是时间序列问题，如果只对单一的分钟数据作分析，会大大减弱数据的时间相关性，因此有必要进行时间序列的组合。将每五分钟的归一化数据（已加入各个特征）组合到一起，将原来的二维数组转变为三维数组进行分析。

（六）提取打标签的依据

对机器学习来说，训练集和测试集都应有对应的标签，然后根据预测值与实际标签值之间的差值进行修正，才能达到提高测试准确率的目的。在 V1.0 版本中，我们只对下一分钟的涨跌情况进行分类，因此标签就是下一分钟的涨跌情况。

首先，查阅资料得知，一个常用的涨跌幅计算方法为：

平均价：MeanPrice = (Latestprice + Minprice + Maxprice + First.Latestprice)/4

下一分钟涨跌幅：（下一分钟平均价 – 当前分钟平均价）/当前分钟平均价

因此，平均价 MeanPrice 和下一分钟涨跌幅 RaiseDown 也加入到数据的特征中去，并且 RaiseDown 直接作为打标签的依据。

（其实，最理想的标签依据是预测下一分钟的涨跌幅,通过计算下一分钟的收盘价和开盘价之间的涨跌幅即(Latestprice – First.Latestprice)/ Latestprice)，这是最理想的预测。因为这样预测很好写策略,但是在实际训练中是很难训练的,效果不好，因为这个涨跌幅和上一分钟没有建立直接关系,想通过训练建立联系比较困难。）

三、划分训练集与测试集

通过以上步骤的数据处理，得到数据集如下所示：

```
array([[[ 1.        ,  1.        ,  1.        , ...,  0.65315817,
          1.        ,  0.51094944],
        [ 0.98349537,  0.9947952 ,  0.98259529, ...,  0.64915289,
          0.98885873,  0.50056431],
        [ 0.97558218,  0.97827563,  0.96959683, ...,  0.64581164,
          0.97517249,  0.54248697],
        [ 0.97354737,  0.97080788,  0.97334215, ...,  0.64486167,
          0.97189232,  0.54957615],
        [ 0.97467782,  0.96922381,  0.97003745, ...,  0.64644872,
          0.97036534,  0.56973569]],

       [[ 0.98349537,  0.9947952 ,  0.98259529, ...,  0.64915289,
          0.98885873,  0.50056431],
        [ 0.97558218,  0.97827563,  0.96959683, ...,  0.64581164,
          0.97517249,  0.54248697],
        [ 0.97354737,  0.97080788,  0.97334215, ...,  0.64486167,
          0.97189232,  0.54957615],
        [ 0.97467782,  0.96922381,  0.97003745, ...,  0.64644872,
          0.97036534,  0.56973569],
        [ 0.97806918,  0.97239194,  0.97554527, ...,  0.65061808,
          0.97381518,  0.54660311]],
```

是一个大小为(35753, 5, 33)的三维数组。

（1）提取训练数据与测试数据

取前面 10000 个数据作为训练集。前面的数据点因有 na 值填充，会对准确性造成一定影响，因此从第 1000 个点开始，到第 11000 个点结束。

将之后的 2000 个数据作为测试集，因为 11000-12000 这一段的价格起伏过于平缓，不宜测试模型准确性，因此选择 12000-14000 部分的 2000 个数据作为测试集。

（2）打标签

根据下一分钟的涨跌幅情况，将数据分为五类，即：大涨、小涨、平稳、小跌、大跌。阈值设定根据大量数据的涨跌幅统计得出，根据本模型设计的阈值，训练数据的五类涨跌幅情况所占的数据数量如下：

```
大涨:  244
小涨:  2136
平稳:  5124
小跌:  2313
大跌:  183
```

大涨和大跌数量较少，平稳数量最多，较符合实际。

根据这个阈值，测试数据的五类涨跌幅情况所占的数据数量如下：

```
大涨:  17
小涨:  320
平稳:  1304
小跌:  349
大跌:  10
```

与训练集相比平稳情况相对较多，但仍在可接受范围内。

至此，训练数据、训练标签、测试数据、测试标签都已准备完毕，均为矩阵形式，大小分别为：(10000, 5, 33)，(10000, 5)，(2000, 5, 33)，(2000, 5)。

四、LSTM 神经网络构建

（一）使用框架描述

本模型的建立基于 TensorFlow 深度学习框架。它的工作模式如下：

· 使用图 (graphs) 来表示计算.
· 在会话 (Session) 中执行图.
· 使用张量 (tensors) 来代表数据,给予固定的格式。
· 通过变量 (Variables) 维护状态,进行赋值、初始化等操作。
· 使用供给 (feeds) 和取回 (fetches) 将数据传入或传出任何操作
· 提供了一系列的计算损失率和最优化的方法,易于调用

（二）典型语句

1.xtr = tf.placeholder("float", [None, n_steps, n_input])
ytr = tf.placeholder("float", [None,n_classes])
占位符，表示输入输出数组的格式。其中 None 表示任意大小。

2.weights = { 'out': tf.Variable(tf.random_normal([n_hidden, n_classes]))}
biases = {'out': tf.Variable(tf.random_normal([n_classes]))}
初始化权值和偏移量，采用随机数赋值。

3.cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(pred, ytr))
optimizer = tf.train.AdamOptimizer(learning_rate=learning_rate).minimize(cost)
代价函数和最优化方法，其中利用交叉熵表示实际输出与预测输出之间的差异，然后使用 adam 最优化方法使得代价函数最小。（另外一个常用的最优化方法是梯度下降法）

4.correct_pred = tf.equal(tf.argmax(pred, 1), tf.argmax(ytr, 1))
accuracy = tf.reduce_mean(tf.cast(correct_pred, tf.float32))
评估模型的准确率

5.sess = tf.InteractiveSession()
acc = sess.run(accuracy, feed_dict={xtr: batch_x, ytr: batch_y})
loss = sess.run(cost, feed_dict={xtr: batch_x, ytr: batch_y})
创建交互式会话并运行，在运行过程中不断输入分段数据，计算准确率和损失度。

五、执行结果

（一）测试准确率

运行模型，能够分段输出训练的准确率和损失度，并且最终输出测试集的分类准确率。如下所示：

```
Iter 6500, Minibatch Loss= 1.412037, Training Accuracy= 0.38000
Iter 7000, Minibatch Loss= 1.066712, Training Accuracy= 0.54000
Iter 7500, Minibatch Loss= 1.789981, Training Accuracy= 0.26000
Iter 8000, Minibatch Loss= 1.216005, Training Accuracy= 0.48000
Iter 8500, Minibatch Loss= 1.204752, Training Accuracy= 0.58000
Iter 9000, Minibatch Loss= 1.209887, Training Accuracy= 0.48000
Iter 9500, Minibatch Loss= 1.231871, Training Accuracy= 0.50000
Optimization Finished!
('Testing Accuracy:', 0.69999999)
```

可见迭代完成后，测试准确率达到了接近 70%，结果较好。

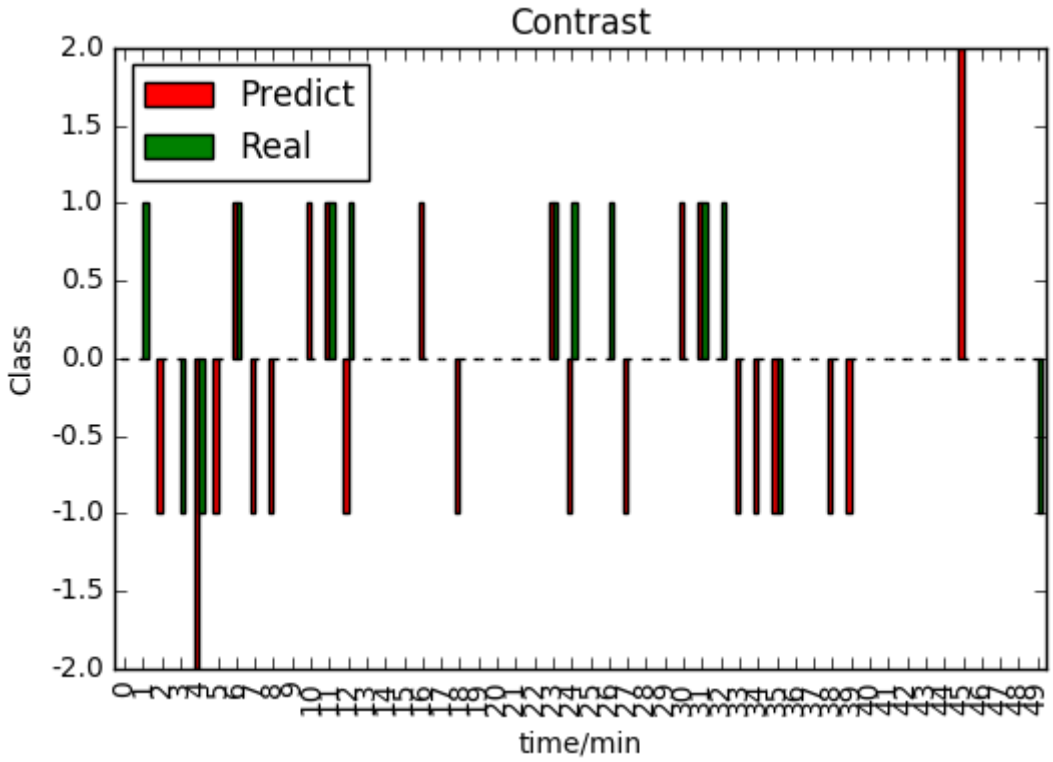
（二）预测类别与实际类别对比

```
Test 0 Prediction: 3 True Class: 2
Test 1 Prediction: 2 True Class: 2
Test 2 Prediction: 3 True Class: 3
Test 3 Prediction: 3 True Class: 3
Test 4 Prediction: 2 True Class: 3
Test 5 Prediction: 3 True Class: 3
Test 6 Prediction: 3 True Class: 2
Test 7 Prediction: 3 True Class: 1
Test 8 Prediction: 1 True Class: 1
Test 9 Prediction: 2 True Class: 2
Test 10 Prediction: 2 True Class: 2
Test 11 Prediction: 2 True Class: 2
Test 12 Prediction: 2 True Class: 2
Test 13 Prediction: 2 True Class: 2
Test 14 Prediction: 2 True Class: 2
```

如上图，执行过程中还可显示每个测试数据的预测类别与实际类别对比。对于大部分测试数据，预测类别和实际类别是一致的。

（三）预测类别与实际类别对比图

为了更直观地进行对比，选取了类别齐全的某 50 分钟数据，作出如下的柱状图：



其中，红色柱状条代表预测类别，绿色柱状条代表实际类别。2.0 高度代表大涨，1.0 高度代表小涨，无高度代表平稳，-1.0 高度代表小跌，-2.0 高度代表大跌。

无柱状区域，以及红绿两条柱状图叠加的点，都是代表预测准确的时间点。有单独柱状的点，代表预测错误的时间点。可见，大部分的点预测准确，其中平稳数据预测的准确性更高。

六、下一步改进方向

该模型还有以下不足之处：

（1）只能根据当前时间预测下一分钟的涨跌情况，还不能根据前面几分钟的数据预测后面几分钟的涨跌情况。这对于策略编写来说还远远不足。

（2）还有一些特征未被加入，如 RSI 指标、布林带指标等，下一步持续加入，以取得更佳的预测效果。

（3）为了更为方便的编写策略，预测的分类应当不只包括下一分钟的涨跌情况，还应该预测上涨趋势（或者下跌趋势）持续的时间，以便更好的确定建仓时间和平仓时间。

（4）对于神经网络的代码，迭代次数过多时会出现程序出错的情况，还没有完全调好。调好之后使用更大的迭代次数应当会取得更好的预测效果。

（Powered by Feng Zipeng）