| 组员 | 论文结构与格式（25） | 论文语言与内容（25） | 课题完成情况（25） | 数据分析与讨论（25） | 总分（100） |
|---|---|---|---|---|---|
| 苏日清 | | | | | |
| 余思贤 | | | | | |
| 梁宗威 | | | | | |
| 谭铭濠 | | | | | |

教师评语：_____

# 基于 GRU 的股市预测


姓名/学号　1、苏日清　2018054439

2、余思贤　2018054439

3、梁宗威　2018054439

4、谭铭濠　2018054439

指导教师：庄师强

电气工程及其自动化


二〇二一年十一月二十七日

# 1 课程设计的任务与要求

## 1.1 课程设计的任务

1. 熟悉 MATLAB 中深度学习工具箱的使用方法，喂入数据的方法，配置训练的方法，保存模型的方法和调用模型的方法；
2. 能画出学习模型的基本框架，理解其基本原理；
3. 基于 GRU 对中国石化的开盘股价进行预测。

## 1.2 课程设计的要求

1. 学会 MATLAB 软件的安装；
2. 熟练掌握 MATLAB 的使用，掌握深度学习工具箱的使用；
3. 能使用深度学习工具箱根据需求搭建神经网络，喂入数据，训练，得出模型并能调用；
4. 通过调参使得模型能更好的贴合实际，打到更好的效果。

# 2 研究基础

## 2.1 序列数据神经网络

### 2.1.1 RNN 网络

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。相比一般的神经网络来说，他能够处理序列变化的数据。

比如某个单词的意思会因为上文提到的内容不同而有不同的含义，RNN 就能够很好地解决这类问题。

一般来说，普通 RNN 网络形式如下：

$$h' = \delta(w_h h + w_i x + b^h) \tag{2.1}$$

$$y = \delta(w_o h' + b^y) \tag{2.2}$$

其中 $h'$ 为传入下一节点的参数，$y$ 常使用对 $h'$ 进行唯独映射，然后使用 $softmax$ 进行分类得到所需的参数。

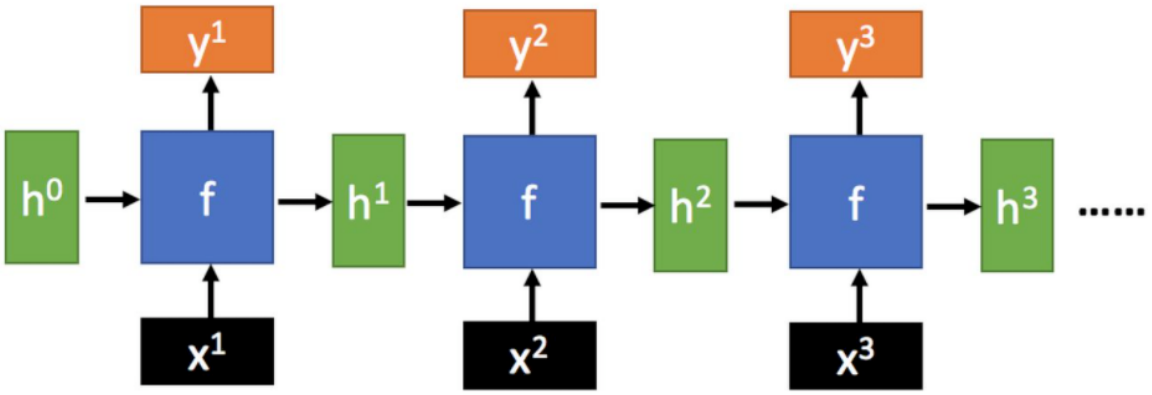


图 2.1: 多级 RNN 网络构成

### 2.1.2 LSTM 结构

长短期记忆（Long short-term memory, LSTM）是一种特殊的 RNN，主要是为了解决长序列训练过程中的梯度消失和梯度爆炸问题。简单来说，就是相比普通的 RNN，LSTM 能够在更长的序列中有更好的表现。

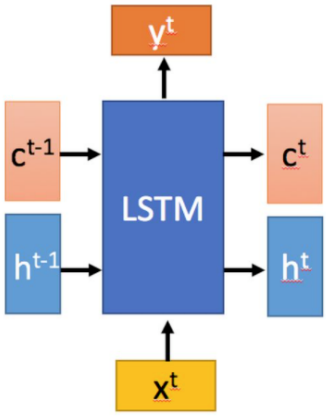


图 2.2: LTSM 网络结构

LSTM 网络的表达式如下：

$$c^t = z^f \odot c^{t-1} + z^i \odot z \tag{2.3}$$

$$h^t = z^o \odot \tanh c^t \tag{2.4}$$

$$y^t = \delta(W' h^t + b^y) \tag{2.5}$$

其中 $c^t$ 可以理解为长期记忆，主要是用来保存节点传递下来的数据的，每次传递会对某些维度进行“忘记”并且会加入当前节点所包含的内容；，$h^t$ 则为短期记忆，仅保存了先前节点的信息。

### 2.1.3 GRU 网络

GRU（Gate Recurrent Unit）是循环神经网络（Recurrent Neural Network, RNN）的一种。和 LSTM（Long-Short Term Memory）一样，也是为了解决长期记忆和反向传播中的梯度等问题而提出来的。

GRU 的输入输出结构与普通的 RNN 是一样的。

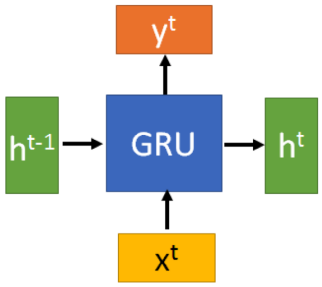


图 2.3: GRU 的输入输出结构

其表达式为：

$$r^t = \delta(x^t w^{xr} + h^{t-1} w^{hr} + b^r) \tag{2.6}$$

$$z^t = \delta(x^t w^{xz} + h^{t-1} w^{hz} + b^z) \tag{2.7}$$

$$h' = \tanh x^t w^{xr} + r^t \odot h^{t-1} w^{hh} + b^h \tag{2.8}$$

$$h^t = (1 - z) \odot h' + z \odot h^{t-1} \tag{2.9}$$

$$y^t = \text{softmax}(h^t w^{hy} + b^y) \tag{2.10}$$

GRU 与 LSTM 相比，需要训练的参数较少，但也能达到与 LSTM 相近的效果。其训练的参数少，对硬件要求要求较低，因此本文采用 GRU 进行实现。

## 2.2 训练数据的准备

### 2.2.1 数据采集

这里使用 Python 爬取近 16 年中国石化（600028）的股市信息。

代码如下：

```
import tushare as ts

df1 = ts.get_k_data('600028', ktype='D', start='2005-01-01', end='2021-10-16')

datapath1 = "./SH600028.csv"
df1.to_csv(datapath1)
```

得到了 4032 行数据，包括：时间、开盘价格、收市价、高位、低位、成交量以及股票代码。

### 2.2.2 数据处理

为使得模型更快收敛，并提高其准确性，对取得的数据进行归一化处理，公式如下：

$$x^* = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2.11}$$

核心代码如下：

```
from sklearn.preprocessing import MinMaxScaler

sc = MinMaxScaler(feature_range=(0, 1))
training_set_scaled = sc.fit_transform(training_set)
test_set = sc.transform(test_set)
```

### 2.2.3 数据标注

为了实现股票预测，需要对训练用的数据进行标注。

GRU 是根据过去预测未来，因此采用前两个月的数据进行对当日开盘价的预测。

取出后 200 日的数据作为测试数据，前 3832 日数据作为训练数据。

使用 Python 进行数据标注，并保存为 txt 文件，方便在 MATLAB 中进行读取调用。

标注核心代码如下：

```
import pandas as pd

test_num=200
day=60

zhongshihua = pd.read_csv('./SH600028.csv')
training_set = zhongshihua.iloc[0:len(zhongshihua) - test_num, 2:3].values
test_set = zhongshihua.iloc[len(zhongshihua) - test_num:, 2:3].values

for i in range(day, len(training_set_scaled)):
    x_train.append(training_set_scaled[i - day:i, 0])
    y_train.append(training_set_scaled[i, 0])
```

## 2.3 股市预测的 Python 实现

先在 Python 中调用 tensorflow 框架进行本项目的实现，验证其可行性。


图 2.4: 股市预测网络结构

核心代码如下（仅展示模型部分）：

```
import tensorflow as tf
from tensorflow.keras.layers import Dropout, Dense, GRU

model = tf.keras.Sequential([
    GRU(512, return_sequences=True),
    Dropout(0.2),
    GRU(1024),
    Dropout(0.2),
    Dense(1)
    ])
```

第一层 GRU：512 个单元，每次返回 $h^t$ 参数；令其中 20% 的单元休眠；第二层 GRU：1024 个单元，仅最后一次返回 $h^t$ 参数；令其中 20% 的单元休眠；最后进行全链接输出。

在 epochs=100，batch size=32 的训练条件下，预测结果如图2.5：

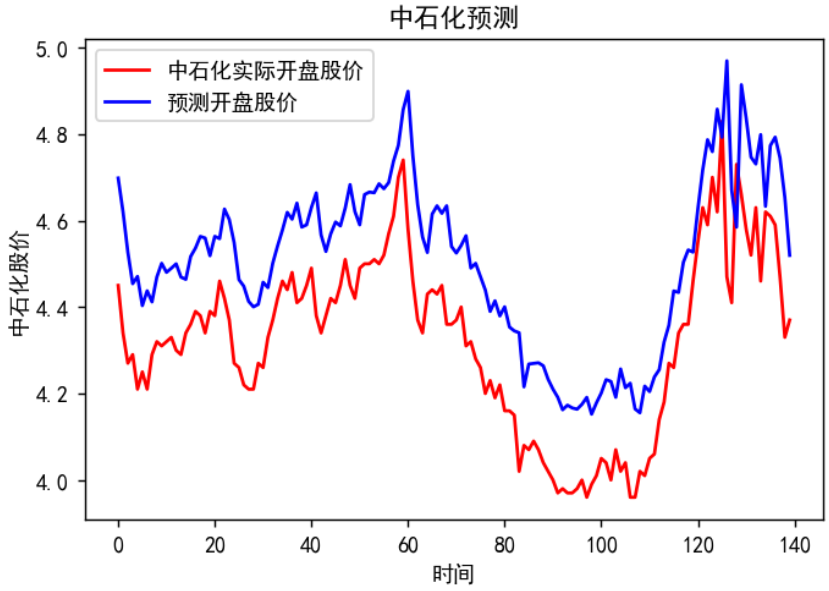


图 2.5: 预测结果

可以看到：预测出的趋势与实际相比是较为准确的。

# 3 股市预测的 MATLAB 实现

## 3.1 深度学习工具箱

这里你们来写，写完发字给我我打进来

## 3.2 在 MATLAB 中搭建 GRU 神经网络

搭一个图

## 3.3 喂入数据并训练

## 3.4 模型优化

### 3.4.1 效果评判

后续测试模型中，我们采用茅台 2005 年 1 月 1 日到 2021 年 3 月 20 日的开盘股价作为训练集，为了节省时间，规定每一种模型均跑 100Epoch。

我们使用如下指标进行模型效果评判：

1. 计算茅台在 2021 年 3 月 30 日到 2021 年 10 月 16 日，共计 200 日的开盘股价预测数据，并与此 200 日的实际开盘股价计算确定系数。

   确定系数计算公式如下：

   $$r(X,Y)=\frac{\mathtt{Cov}(X,Y)}{\sqrt{\mathtt{Var}[X]\mathtt{Var}[Y]}} \tag{3.1}$$

   式中，$\mathtt{Cov}(X,Y)$ 为 X 与 Y 的协方差，$\mathtt{Var}[X]$ 为 X 的方差，$\mathtt{Var}[Y]$ 为 Y 的方差。

2. 计算模型计算速度，测试数据为：茅台在 2021 年 3 月 30 日到 2021 年 10 月 16 日的股价预测数据。计算速度测试中，时间计算单位为 ms，一共跑 100 次，取平均值。

3. 测试设备为联想拯救者 R9000P（5800H+64G 内存 +RTX3070-8G）

4. 综合评判公式：

   **先这么写着先，我并不确定这个可不可行**

   $$s=\frac{r*10+T}{r*T} \tag{3.2}$$

### 3.4.2 优化方式

#### 3.4.2.1 更改模型

在 3.3 中，我们得到了粗略的结果，如图2.5，其模型如图2.4。从结果中可以看出，虽然预测所得的结果趋势大致与实际情况近似，但其偏差还是较大。因此，在这一节中，我们更改了模型。其中更改的方向如下：

1. 增减模型层数

2. 调整 Dropout 单元的数量和 Dropout 的数值

3. 调整 GRU 内神经元个数

4. 在 GRU 中使用 sigmond 替换 tanh

5. 调整 beatch-size 都大小

我们据此作出了 10 个新的模型，如表3.1所示。

表 3.1: 10 种调参模型

| 序号 | 模型结构 | 备注 |
|---|---|---|
| 1 | GRU(20)→Dropout(0.2)→GRU(20)→Dropout(0.2)→Dense | |
| 2 | GRU(20)→Dropout(0.2)→GRU(20)→GRU(20)→Dropout(0.2)→Dense | |
| 3 | GRU(40)→Dropout(0.2)→GRU(80)→Dropout(0.2)→Dense | |
| 4 | GRU(20)→Dropout(0.5)→GRU(20)→Dropout(0.5)→Dense | |
| 5 | GRU(80)→Dropout(0.2)→GRU(160)→Dropout(0.2)→Dense | |
| 6 | GRU(20)→GRU(20)→Dropout(0.2)→Dense | |
| 7 | GRU(20)→Dropout(0.2)→GRU(20)→Dropout(0.2)→Dense | 输出使用 sigmoid |
| 8 | GRU(20)→Dropout(0.7)→GRU(20)→Dropout(0.7)→Dense | |
| 9 | GRU(20)→Dropout(0.2)→GRU(20)→Dropout(0.2)→GRU(20)→Dropout(0.2)→Dense | |
| 10 | GRU(20)→Dropout(0.2)→GRU(20)→GRU(20)→GRU(20)→Dropout(0.2)→Dense | |

下面进行模型效果评判，结果如下：

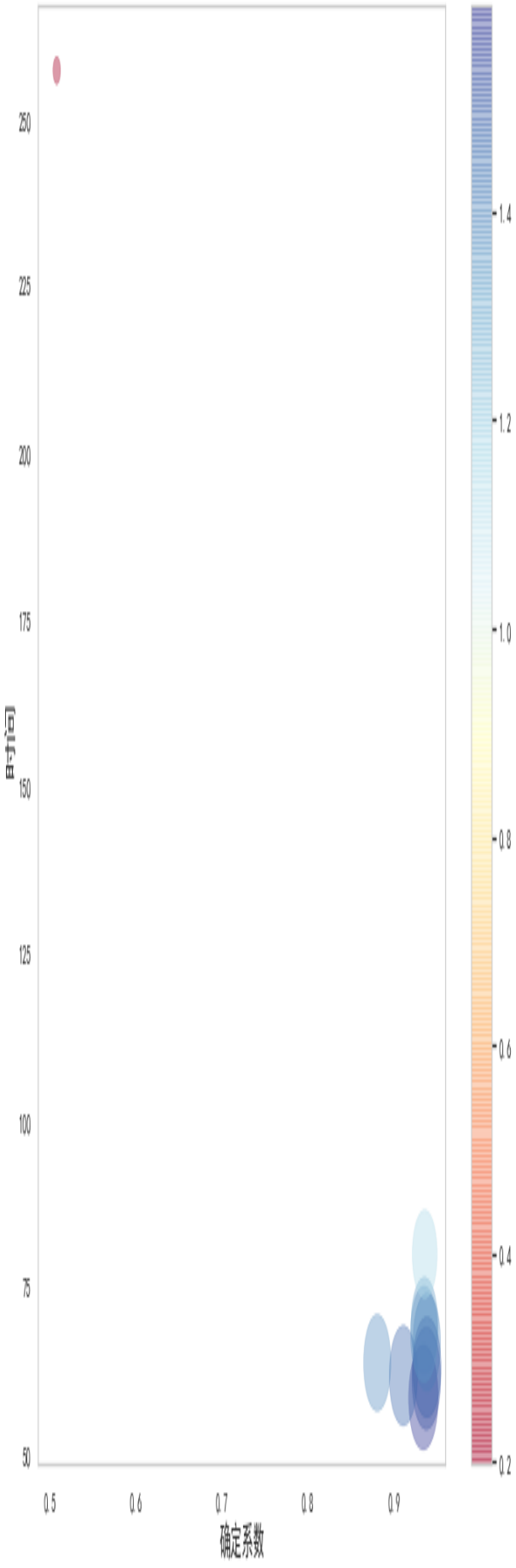
250
225
200
175
150
125
100
75
50
0.5
0.6
0.7
0.8
0.9
确定系数
1.4
1.2
1.0
0.8
0.6
0.4
0.2

图 3.1

#### 3.4.2.2 每个 GRU 单元中偏置 b 初始化为 1[1]

由上一小节，我们得出：模型???，即 xxxxx 的效果最好，因此此后都在此基础上修改。

#### 3.4.2.3 全链接层使用 Highway Network 代替 [2]

#### 3.4.2.4 调整 batch size 大小

### 3.5 结果

# 4 实验总结

# 参考文献

# 参考文献

[1] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350. PMLR, 2015.

[2] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.