1小时预测1小时（5-1~7-16）

by cn

数据每一行包括 Num，Hour，Date，DateType

删除了0点的噪声数据，同时，将每天的小时数定为6时至23小时（保存了客流量为0的情况存在，时段为23时）

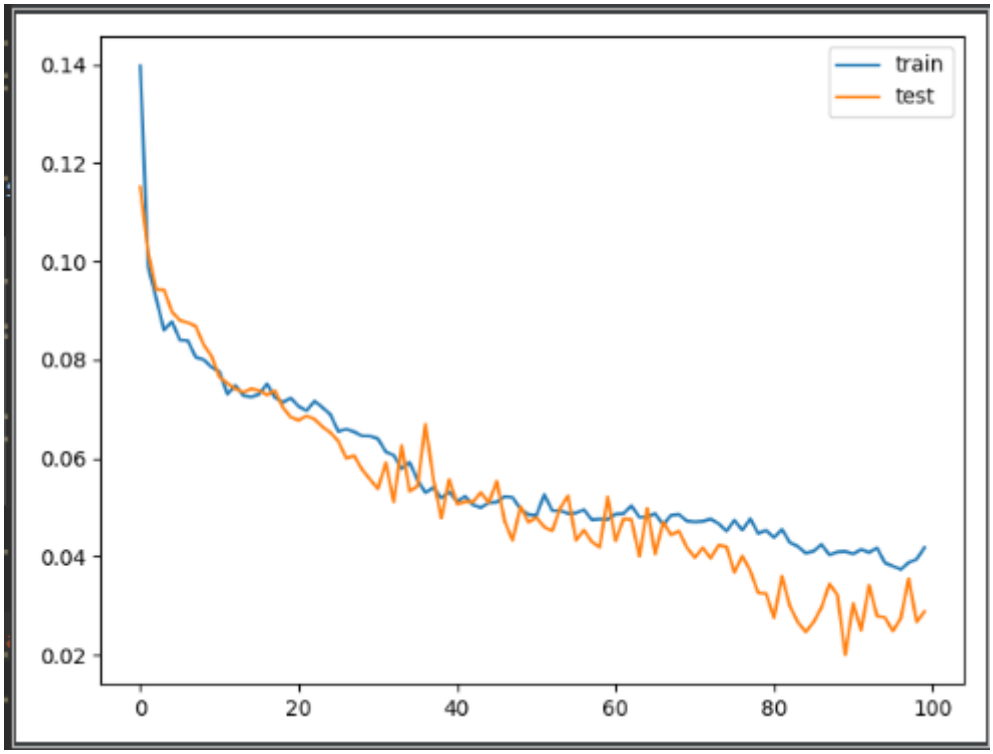
模型如下：

```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

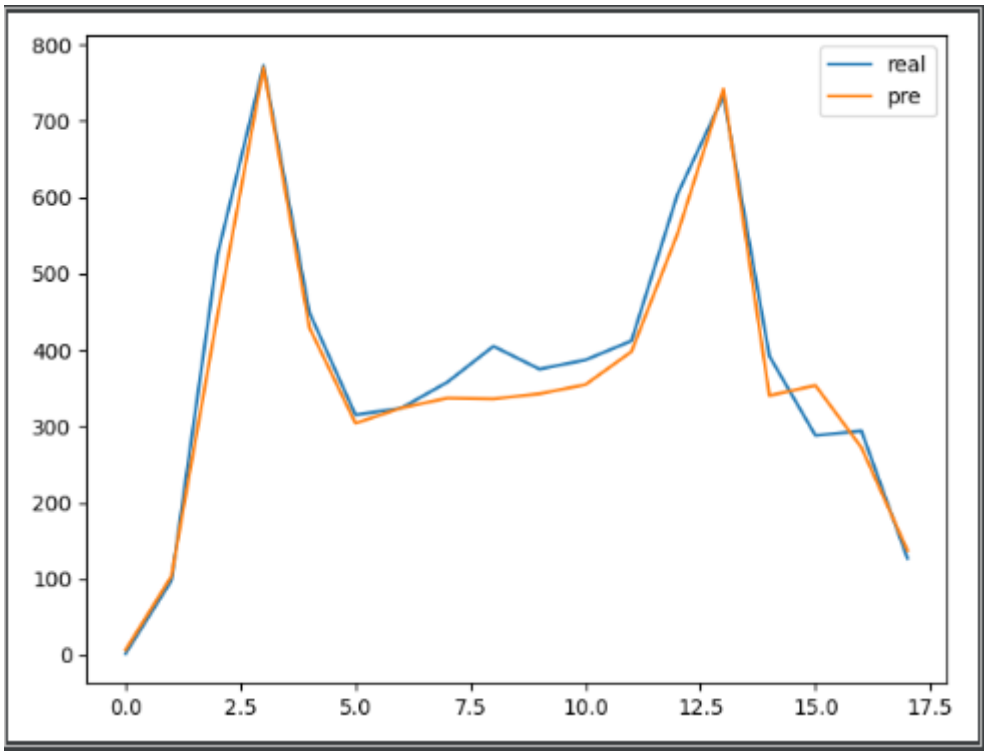
最后的拟合优度和mae如下：

```
Epoch 98/100
 - 1s - loss: 0.0388 - r2: 0.8813 - val_loss: 0.0355 - val_r2: 0.9440
Epoch 99/100
 - 1s - loss: 0.0394 - r2: 0.8850 - val_loss: 0.0267 - val_r2: 0.9627
Epoch 100/100
 - 1s - loss: 0.0418 - r2: 0.8703 - val_loss: 0.0288 - val_r2: 0.9630
Test RMSE: 37.043
```

损失函数如图：



拟合图像：



评估：拟合得较好，仍然存在上升的空间

前N个小时预测一个小时（前n个数据）（5-1~7-16)

数据每一行包括Num，Hour，Date，DateType

数据集与一小时预测一小时的数据集相同

不同的在于这次用了前N个数据进行预测，这里选用的N为18，即前一天6-23小时的数据

```
n_hours = 18
n_features = len(scaled.columns)
```

使用的模型与上面相同：

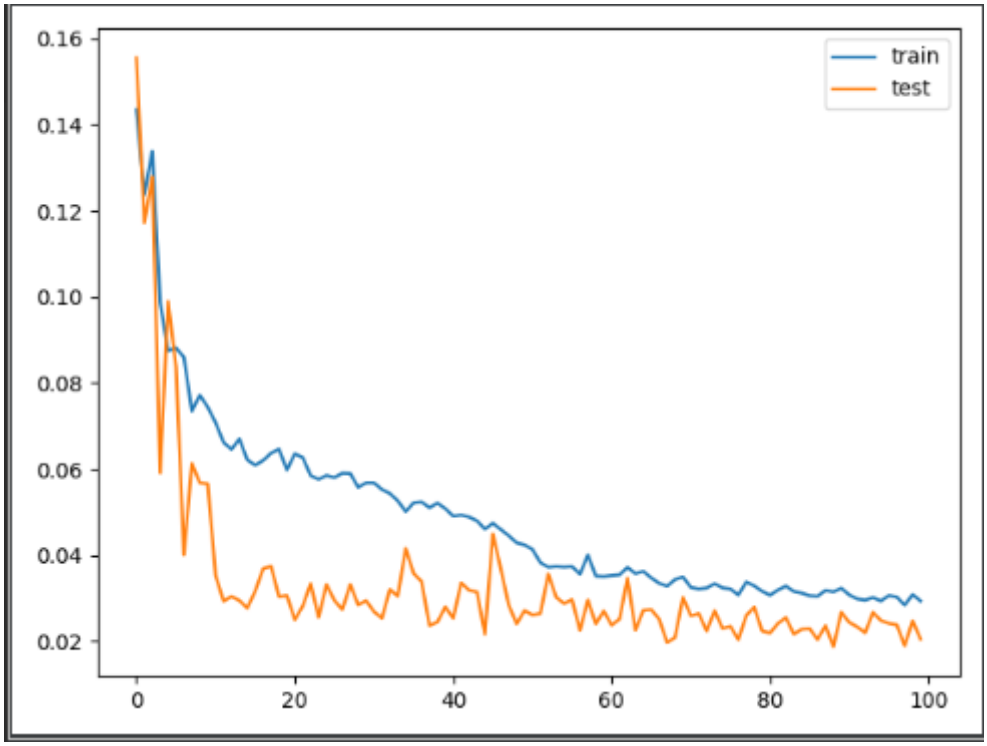
```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

1.预测最后两天

最后的拟合优度和mae如下：

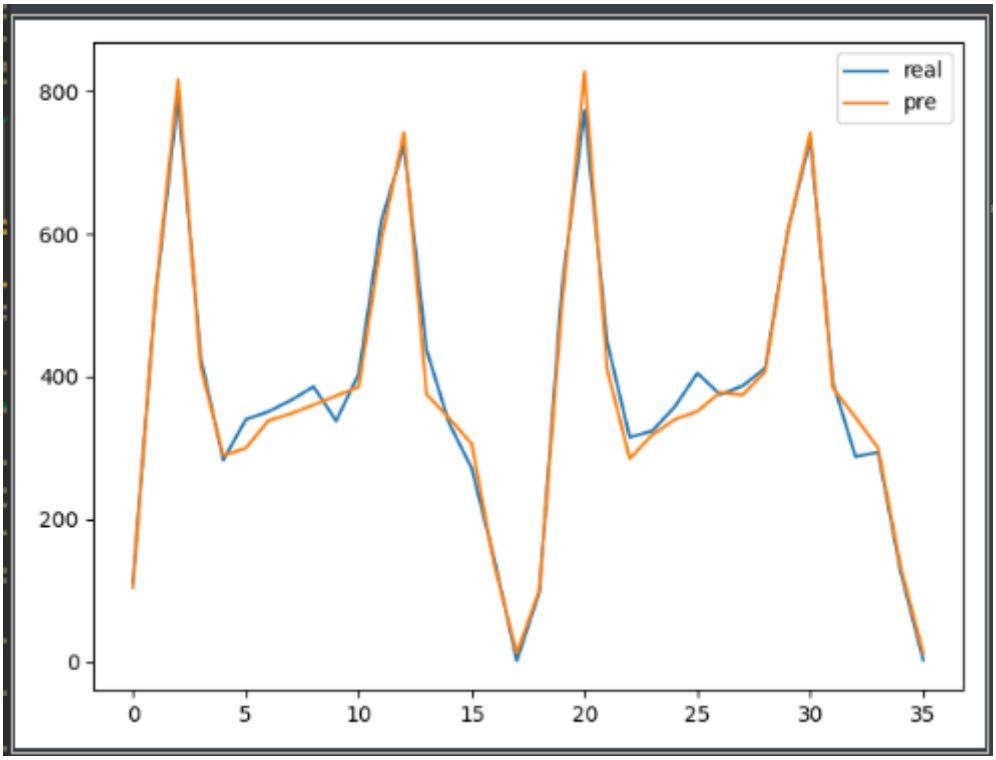
```
Epoch 98/100
 - 4s - loss: 0.0284 - r2: 0.9334 - val_loss: 0.0190 - val_r2: 0.9842
Epoch 99/100
 - 4s - loss: 0.0309 - r2: 0.9272 - val_loss: 0.0248 - val_r2: 0.9754
Epoch 100/100
 - 4s - loss: 0.0294 - r2: 0.9341 - val_loss: 0.0205 - val_r2: 0.9813
```

损失函数：



损失函数	数值
RMSE	25.995
R2	0.982
MAE	19.937
MAPE	0.287

预测图像：



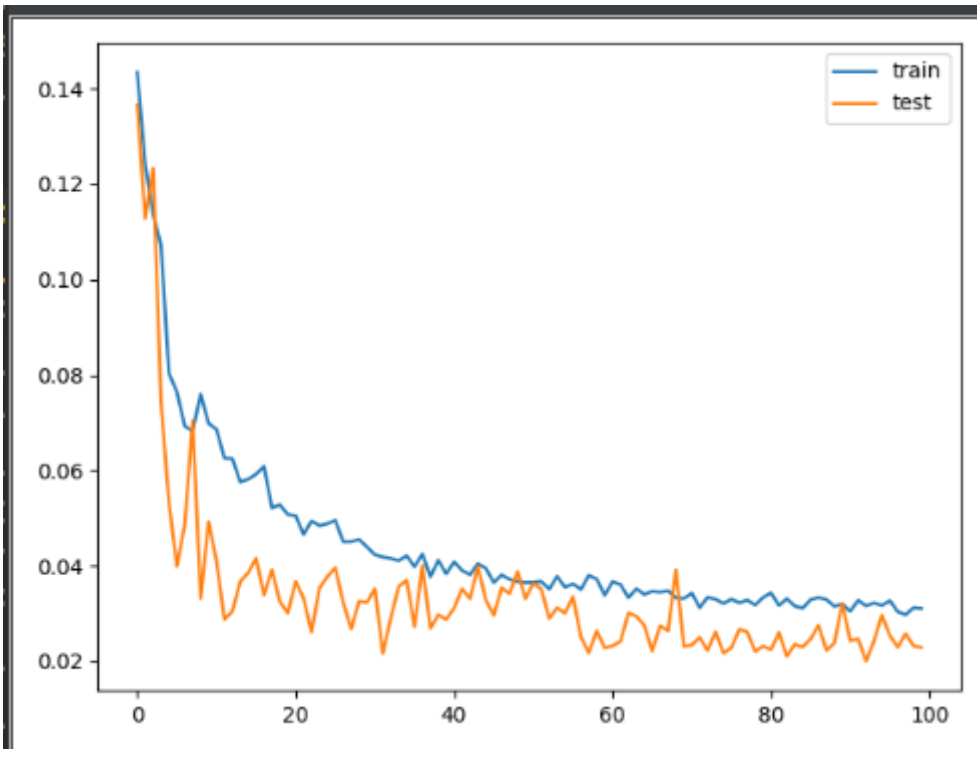
评估：误差比上面的更小

2.预测最后一天

最后的拟合优度和mae如下：

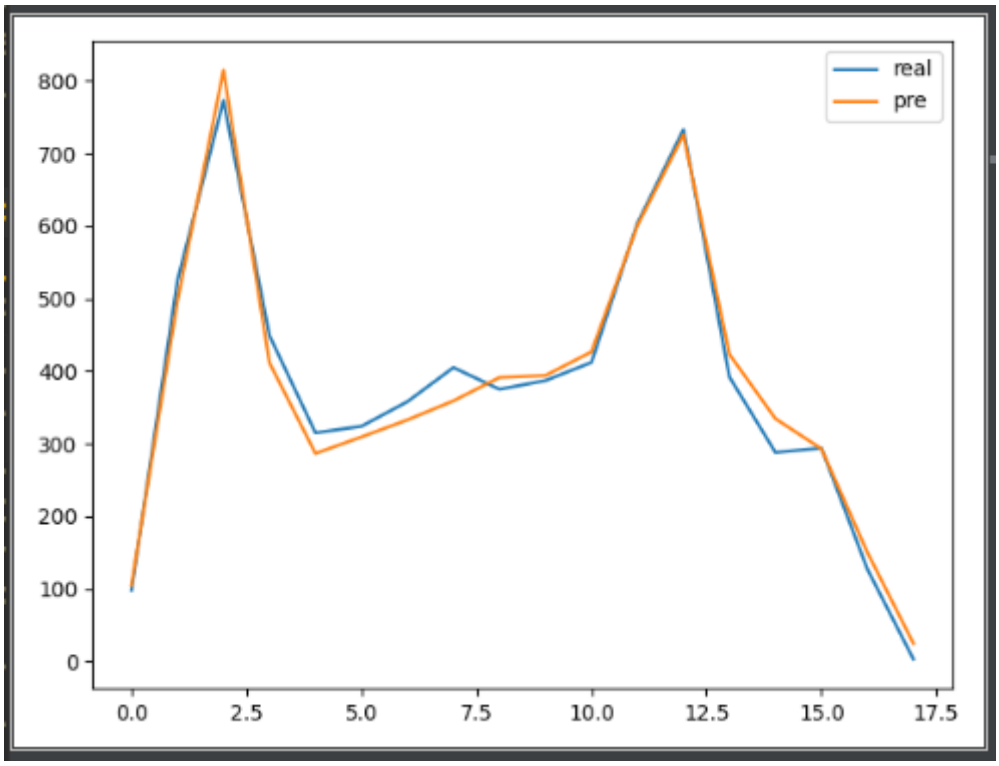
```
Epoch 98/100
 - 9s - loss: 0.0298 - r2: 0.9251 - val_loss: 0.0258 - val_r2: 0.9767
Epoch 99/100
 - 9s - loss: 0.0312 - r2: 0.9164 - val_loss: 0.0232 - val_r2: 0.9821
Epoch 100/100
 - 9s - loss: 0.0311 - r2: 0.9185 - val_loss: 0.0229 - val_r2: 0.9813
```

损失函数：



损失函数	数值
RMSE	26.335
R2	0.981
MAE	22.235
MAPE	0.465

预测图像：



评估：预测的也十分好，十分拟合

前N天的相同时段预测一个小时（5-1~7-16）

数据每一行包括Num，Hour，Date，DateType

数据集与一小时预测一小时的数据集相同

不同的在于这次用了前N天相同时段的数据进行预测，这里选用的N为5，即前五天相同时段的数据

```
n_hours = 5
n_features = len(scaled.columns)
```

使用的模型与上面相同：

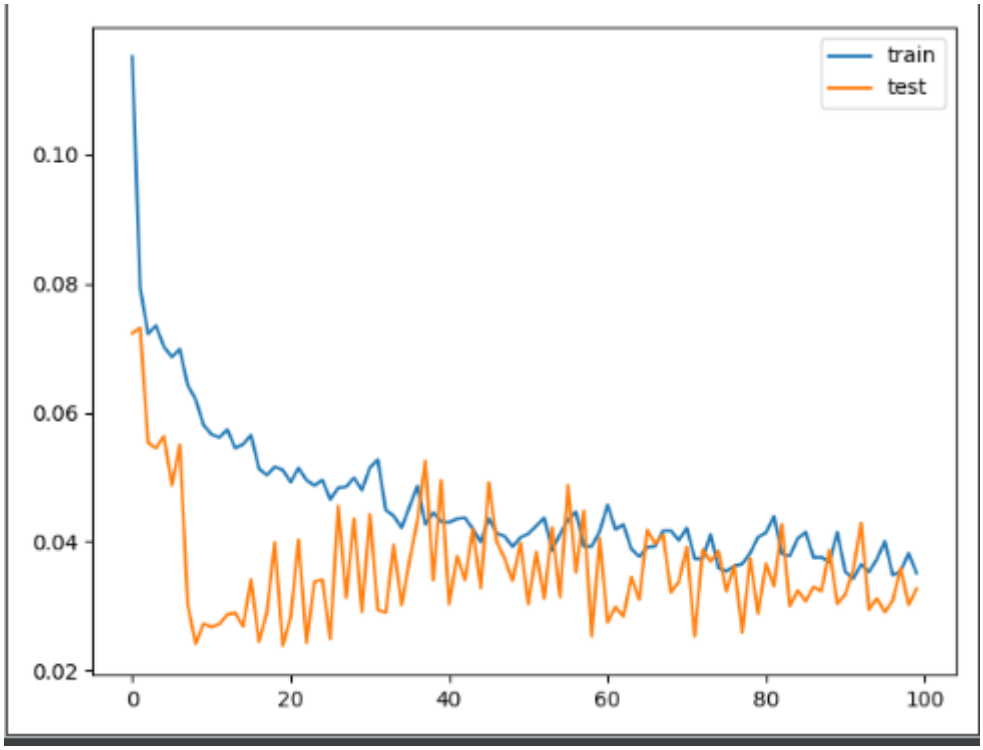
```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

最后的拟合优度和mae如下：

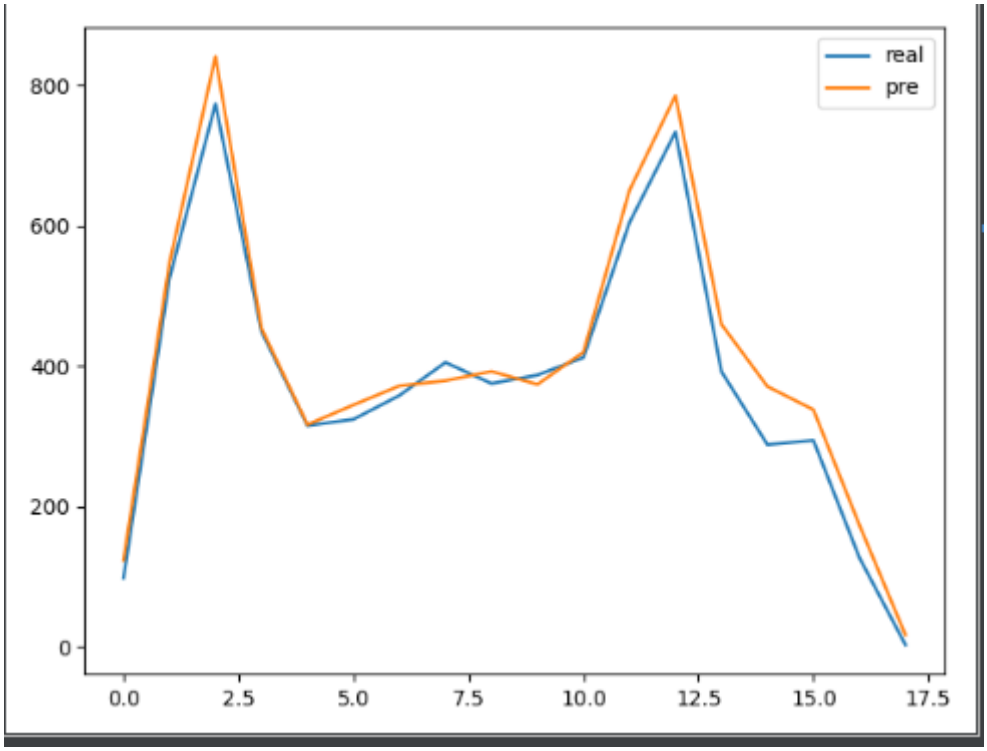
```
Epoch 98/100
 - 2s - loss: 0.0354 - r2: 0.8755 - val_loss: 0.0359 - val_r2: 0.9425
Epoch 99/100
 - 2s - loss: 0.0382 - r2: 0.8734 - val_loss: 0.0301 - val_r2: 0.9594
Epoch 100/100
 - 2s - loss: 0.0351 - r2: 0.8971 - val_loss: 0.0326 - val_r2: 0.9581
```

```
Test RMSE: 39.381
```

损失函数：



拟合图像：



评估：误差反而较大

前N个小时预测一个小时（前n个数据）（1-12~1-23）

数据每一行包括Num，Hour，Date，DateType

数据集只取了1-12~1-23中的数据

不同的在于这次用了前N个数据进行预测，这里选用的N为18，即前一天6-23小时的数据

```
n_hours = 18
n_features = len(scaled.columns)
```

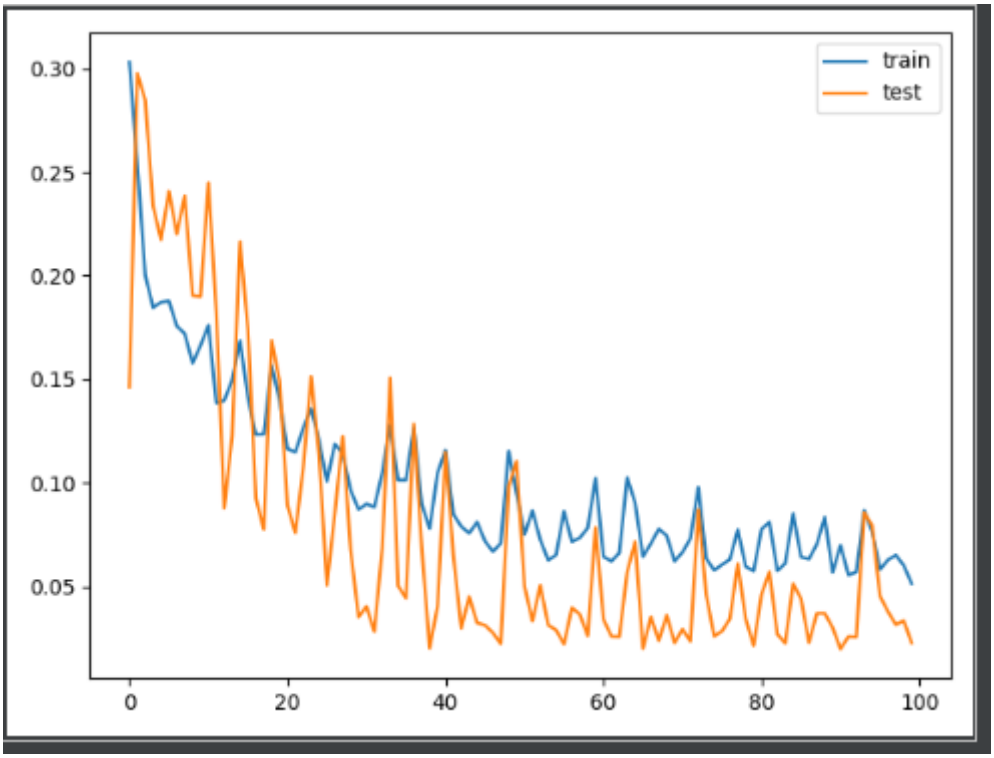
使用的模型与上面相同：

```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

最后的拟合优度和mae如下：

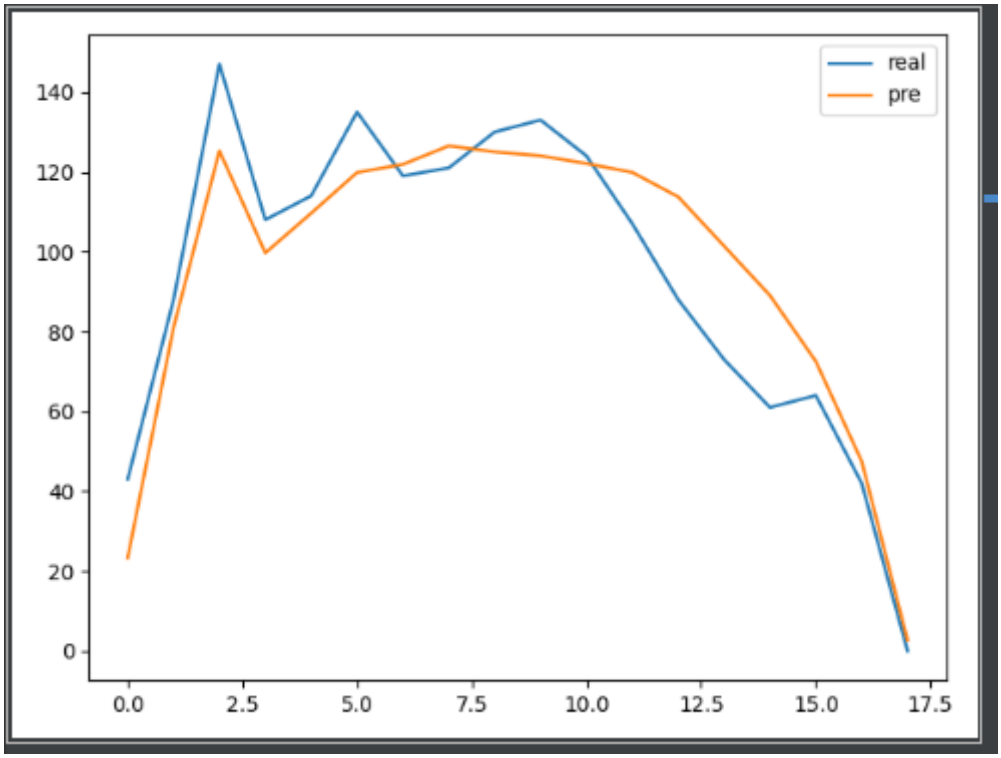
```
Epoch 98/100
 - 1s - loss: 0.0651 - r2: 0.8478 - val_loss: 0.0316 - val_r2: 0.7269
Epoch 99/100
 - 1s - loss: 0.0601 - r2: 0.8644 - val_loss: 0.0334 - val_r2: 0.6805
Epoch 100/100
 - 1s - loss: 0.0513 - r2: 0.8953 - val_loss: 0.0227 - val_r2: 0.8551
```

损失函数：



损失函数	数值
RMSE	14.733
R2	0.855
MAE	11.791
MAPE	inf

预测图像：



评估：不知怎么说...真实值的数据也感觉有点问题，1-23日也比较特殊，可能效果不是很好，还需要想别的办法。

前N个小时预测一个小时（前n个数据）（1-23~3-1）

数据每一行包括Num，Hour，Date，DateType

数据集为1-23~3-1的数据，但数据量较小

不同的在于这次用了前N个数据进行预测，这里选用的N为18，即前一天6-23小时的数据

```
n_hours = 18
n_features = len(scaled.columns)
```

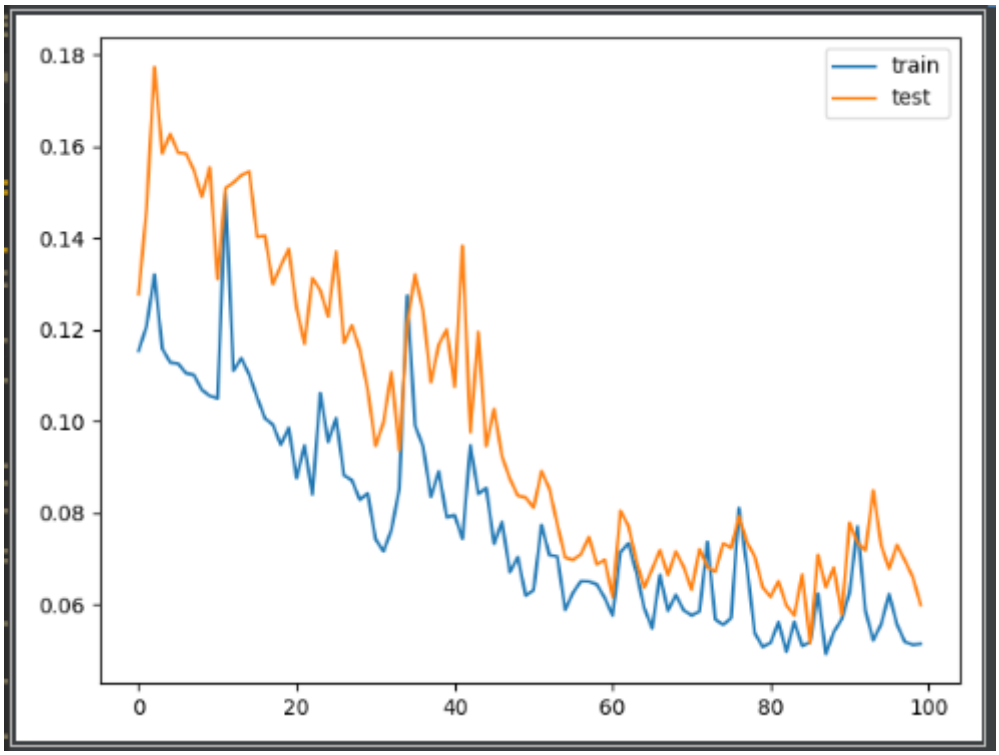
使用的模型与上面相同：

```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

最后的拟合优度和mae如下：

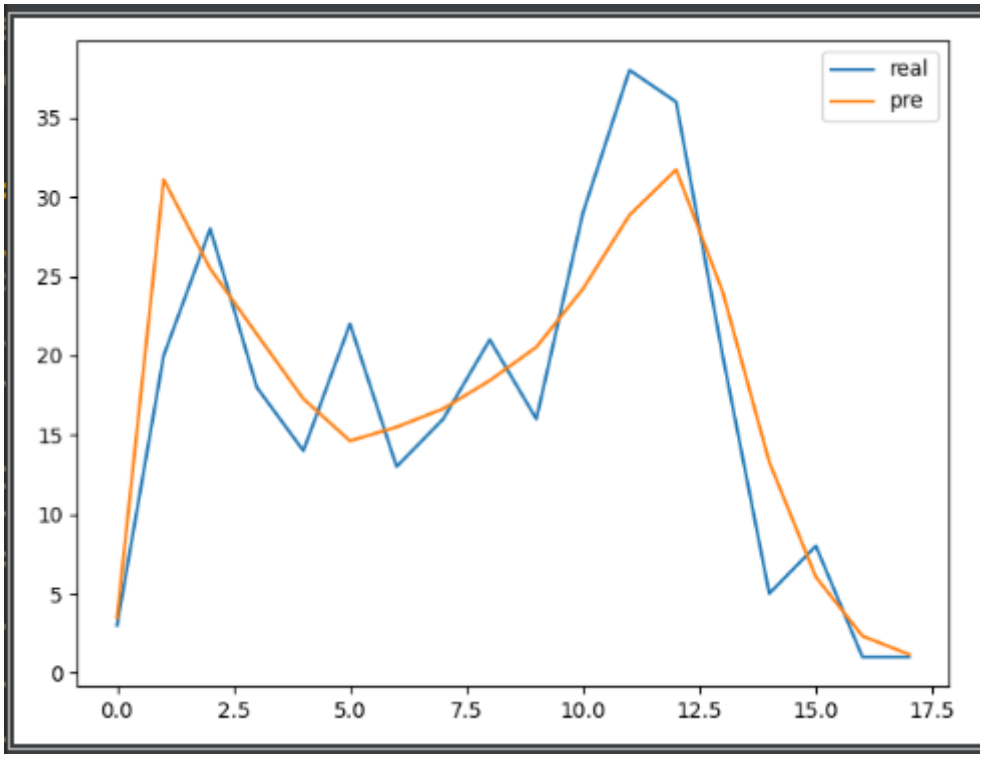
```
Epoch 98/100
 - 2s - loss: 0.0519 - r2: 0.5945 - val_loss: 0.0696 - val_r2: 0.6703
Epoch 99/100
 - 2s - loss: 0.0513 - r2: 0.6194 - val_loss: 0.0660 - val_r2: 0.7397
Epoch 100/100
 - 2s - loss: 0.0515 - r2: 0.6286 - val_loss: 0.0600 - val_r2: 0.7819
```

损失函数：



损失函数	数值
RMSE	5.029
R2	0.782
MAE	4.018
MAPE	0.351

预测图像：



评估：由于数据量较小，拟合得图线较差

前N个小时预测一个小时（前n个数据）（3-1~5-1）

据每一行包括Num，Hour，Date，DateType

数据集取得是3-1~5-1的数据

不同的在于这次用了前N个数据进行预测，这里选用的N为18，即前一天6-23小时的数据

```
n_hours = 18
n_features = len(scaled.columns)
```

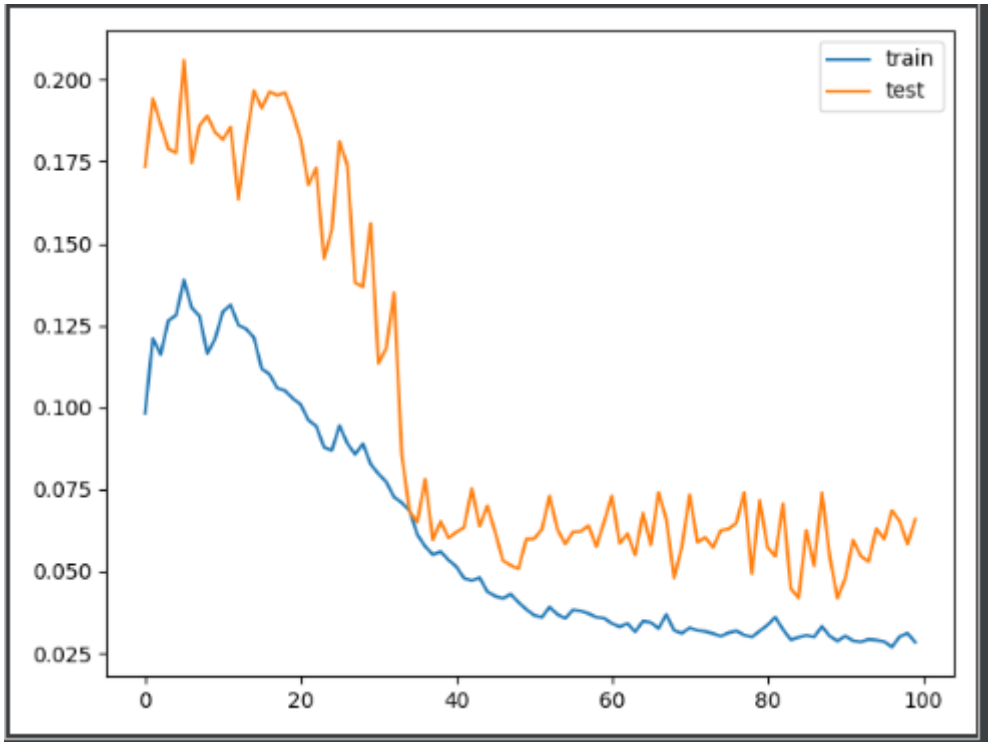
使用的模型与上面相同：

```
# 设计网络结构
model = Sequential()
model.add(LSTM(256, activation="relu",input_shape=(train_X.shape[1], train_X.shape[2]),return_sequences=True))
# model.add(LSTM(256, activation="relu",return_sequences=True))
model.add(LSTM(256, activation="relu"))
model.add(Dropout(0.3))
# model.add(Dense(128))
model.add(Dense(16))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam',metrics=[r2])
# 拟合网络
history = model.fit(train_X, train_y, epochs=100, batch_size=20, validation_data=(test_X, test_y), verbose=2, shuffle=False)
model.save("testmodle.h5")
```

最后的拟合优度和mae如下：

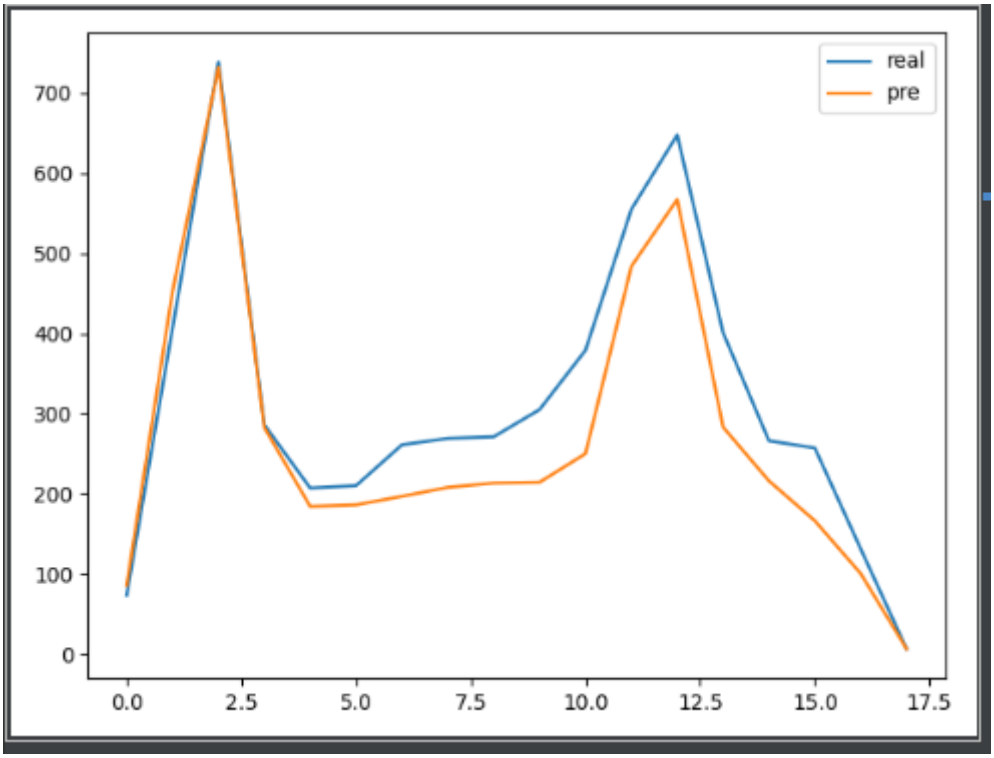
```
Epoch 98/100
 - 4s - loss: 0.0301 - r2: 0.8172 - val_loss: 0.0653 - val_r2: 0.8668
Epoch 99/100
 - 4s - loss: 0.0311 - r2: 0.7702 - val_loss: 0.0582 - val_r2: 0.8909
Epoch 100/100
 - 5s - loss: 0.0284 - r2: 0.8098 - val_loss: 0.0659 - val_r2: 0.8705
```

损失函数：



损失函数	数值
RMSE	65.289
R2	0.870
MAE	53.344
MAPE	0.178

预测图像：



评估：3-1~5-1的数据呈现一种上升的趋势。该模型对峰值的预测较好，能较准确的预测出高峰值，但仍有上升空间。