**机器学习实验报告**

张顾潇 20244227046

2025.1.13

# 问题介绍

共享单车系统革新了传统的自行车租赁服务，实现了从用户注册、租赁自行车到归还的全 流程高度自动化。借助这一系统，用户可以方便地在指定站点租借自行车，并在任意站点归还。截至目前，全球已有超过500个共享单车项目，共计投放了50多万辆自行车。由于共享单车在缓解交通拥堵、改善环境以及促进公众健康等方面发挥了重要作用，越来越受到社会各界的关注和欢迎。

## 1.1 背景和动机

对共享单车租赁数量进行预测可以帮助运营商合理规划和优化车辆调度，提升运营效率和服务质量。 而共享单车租赁需求与环境和季节高度相关。例如，天气条件、降水、星期几、季节、一天中的时间段等都会影响租赁行为。如果能合理预测，运营商能在短期内准确预判租赁需求，提前调配、投放共享单车，避免“供给不足”或“车辆堆积”的情况；同时也能帮助运维部门提高管理效率、缩减成本，并提升用户满意度。

## 1.2 数据和来源

数据来源是美国华盛顿的两年历史骑行数据，和对应时段的天气信息。原始日志记录每天每小时的租赁总量，以及当时的天气、节假日、工作日等条件，可能反映出租赁数量的变化的。数据包含时间类特征：季节（season）、节假日（holiday）、是否工作日（workingday）、星期几（weekday）、小时（hr）等。天气类特征：天气情况（weathersit）、温度（temp）、体感温度（atemp）、湿度（hum）、风速（windspeed）等。

## 1.3 目标和要求

构建能够多步预测的时间序列模型：基于LSTM或Transformer深度学习模型，对过去 96 小时的特征进行学习和建模，预测未来 96和 240小时的共享单车租借总量cnt；在测试集上，通过 MSE、MAE指标评估预测误差，并多次实验计算标准差反应模型稳定性，最后画曲线图与现实数据进行对比。

## 1.4 解决思路

1. **数据处理与滑动窗口构建：**按照时间顺序构建输入序列X和输出序列y的滑动窗口，将原始数据按天、小时进行清洗、特征提取与归一化，形成适合深度学习模型输入的数据格式；
2. **设计和使用模型：**构建 LSTM、Transformer 序列模型，提出的新模型；
3. **训练模型：**模型训练，选择合适的参数；
4. **评估模型：**在测试集上验证模型性能，计算MSE、MAE，以及标准差，同时可视化预测曲线与真实曲线的对比。

# 模型

## 2.1 LSTM

长短期记忆网络（LSTM, Long Short-Term Memory） 是一种常用的循环神经网络（RNN）变体，专门为了解决 RNN 在长序列上训练时可能出现的梯度消失或梯度爆炸问题。它通过在内部维护“输入门、遗忘门、输出门”等机制，对历史状态进行一定程度的“记忆”或“遗忘”，从而在长时间跨度上仍能较好地学习时序依赖。

在本次实验中，LSTM 用于多步时间序列预测：给定过去I=96小时的特征数据（温度、湿度、天气、节假日、工作日、总租借量等），一次性预测未来O=96和O=240小时的共享单车租借量。

### 2.1.1 LSTM 原理简述

LSTM 保持一条贯穿始终的细胞状态Cell State，可以在不被大幅修改的情况下长距离传递信息。

输入门、遗忘门、输出门：

* **输入门**决定当下时刻新信息如何写入细胞状态；
* **遗忘门**决定历史信息如何在细胞状态中被保留或遗忘；
* **输出门**决定当前时刻要输出多少信息到隐藏状态。

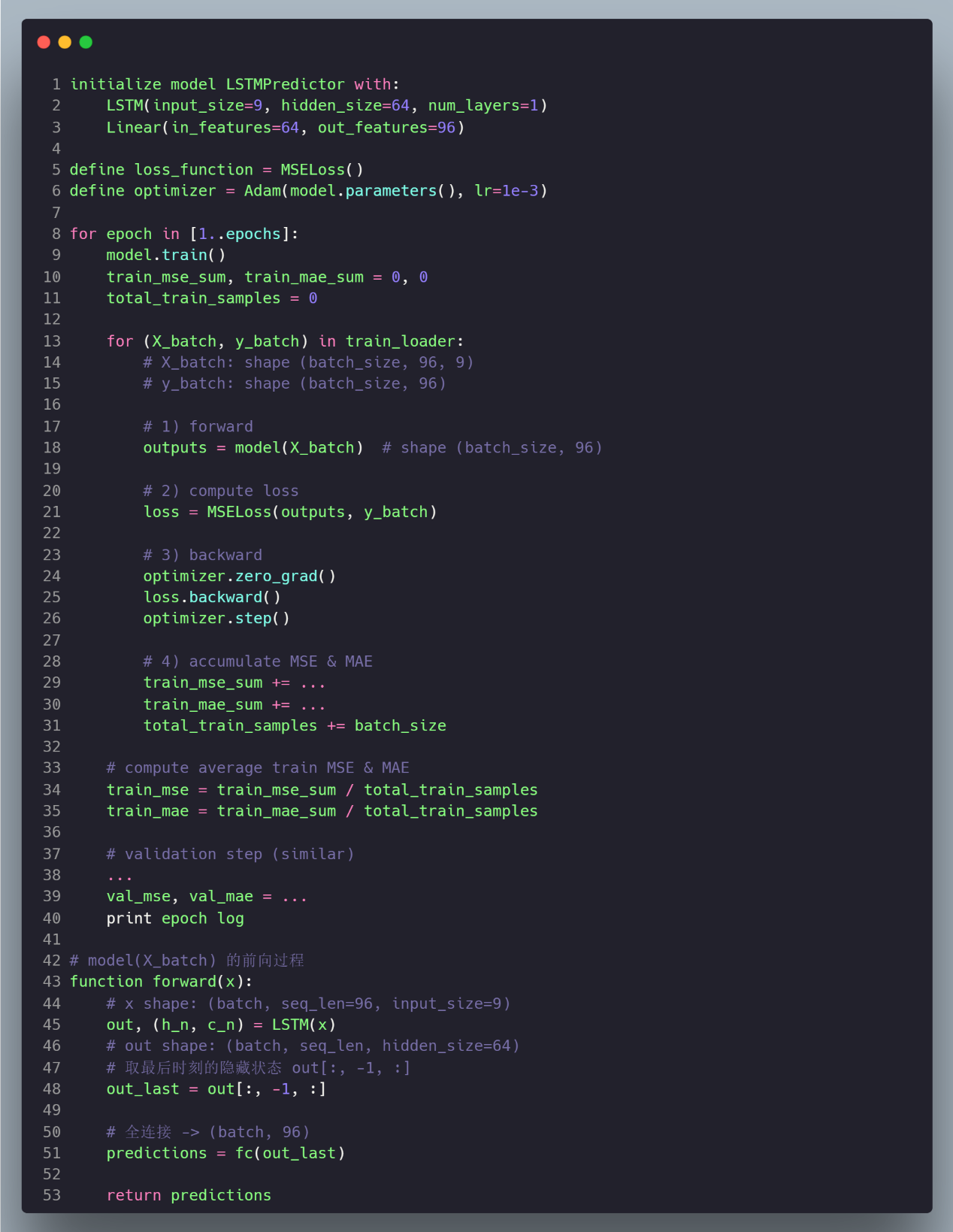
通过以上门控机制，LSTM 能够在面对长时间序列时依然保持稳定的梯度传递。

### 2.1.2 LSTM结构

本实验使用的LSTM结构比较简洁：

1. **输入层：**batch, seq\_len, input\_size，其中input\_size=9即特征列数；
2. **LSTM 层**：隐藏单元数（hidden\_size）=64，能够处理长度为 seq\_len=96 的序列；
3. **全连接层**：将LSTM最后时刻的隐状态映射到输出维度 output\_size=96一次性预测未来96小时或者output\_size=240预测未来240小时。

### 2.1.3 伪代码



### 2.1.3. 训练与评估

1. **数据预处理：**

* 将数据按时间顺序构建“滑动窗口”，即 “过去96小时” -> “未来96小时”。
* 归一化特征temp, hum, cnt等范围到[0,1]。

1. **批量训练：**

* 使用DataLoader将(X, y)分批输入LSTM。
* 用MSE、MAE、Loss评估损失并反向传播。

1. **验证集性能：**

* 每个epoch 结束后，在验证集上做一次前向预测，计算 MSE、MAE并观察是否过拟合，从而确定合适的epoch。

1. **最终测试：**

* 使用最后训练好的模型，在test\_data测试数据上执行同样的滑动窗口预测，评估其 MSE、MAE并可视化预测曲线。

1. **多次训练和测试**

* 通过多次训练和测试，评估模型稳定性。

## 2.2 Transformer

Transformer模型最早由Google的《Attention is All You Need》论文提出，用于机器翻译等序列到序列任务。它摆脱了RNN的循环结构，通过多头自注意力机制并行地捕捉序列内部的长依赖关系。如今在自然语言处理和时间序列预测等领域都获得了大量应用。

### 2.2.1 Transformer时间序列预测

**优势**：可以并行处理序列中的各个位置，且对于长序列时不会像 RNN 那样在时间维度上累积误差。自注意力机制可以更灵活地“关注”远距离信息。

**难点：**时间序列通常需要显式位置编码来保留时序信息。在长序列上，Transformer 可能有计算和内存开销较大的问题。

### 2.2.2 本实验的Transformer结构

1. **输入投影**：将每个时刻的多维特征9维映射到d\_model空间。
2. **位置编码：**使用正余弦位置编码让模型区分时间步顺序。
3. **Encoder：**多层 Self-Attention + FeedForward，对过去 96 小时进行编码。
4. **Decoder：**以全零构造 96 个步长的“目标占位”向量，逐步解码得到未来 96 小时的预测。
5. **输出层：**投影到 1 维，用于预测共享单车租借数量 cnt。

### 2.2.3 Transformer伪代码



### 2.2.4 Transformer训练与评估

1. **位置编码：**

* 为了让Transformer知道输入序列中每个时间步的位置，输入增加正余弦位置编码。

1. **Decoder输入：**

* 采用“全零”序列+96大小的特征维度来让Transformer Decoder解码O=96步。

1. **多步预测：**

* 和LSTM相同，最终输出(batch, O)。

1. **训练与验证：**

* 同样使用MSE、MAE、Loss评估。

1. **测试集评估：**

* 同 LSTM，把滑动窗口应用到测试数据，每个样本调用model(src, tgt\_zeros)，预测出96小时和240小时的序列并反归一化得到实际cnt，计算MAE、MSE并画出曲线图。

1. **稳定性测试：**

* 同LSTM，多次执行测试评估模型再测试集上预测结果的MAE、MS以及稳定性。

## 2.3 新模型：多尺度卷积 + LSTM + 注意力 = MSCLA

下面提出一种融合多尺度卷积（Multi-Scale CNN）、LSTM和注意力机制Attention的混合模型MSCLA，用于多步时间序列预测：给定过去I=96小时的特征数据：温度、湿度、天气、节假日、工作日、总租借量等，预测未来O=96或O=240 小时的共享单车租借量。

### 2.3.1 MSCLA原理

在长序列预测中，单纯的LSTM可能无法充分利用不同时间尺度下的局部信息，尤其是当数据本身存在季节性、周期性或突发峰值等复杂模式时。可以借鉴Temporal Convolutional Network[1]和WaveNet [2]提出的扩大卷积思路，在输入层并行若干条不同膨胀率dilation或不同卷积核大小的卷积分支，提取多尺度的时序特征，然后再将多尺度卷积输出拼接后输入到LSTM以捕捉长时依赖；最后可以在LSTM输出序列上施加注意力机制，让模型自动学习关键时间步对预测的贡献度。

具体流程如下：多尺度卷积层通过并行的不同膨胀率（dilation）或不同卷积核大小的卷积层，提取序列中的多尺度局部特征。多条并行的1D卷积分支，每条分支具有不同的膨胀率或卷积核大小。每条分支输出固定数量的特征通道。最终将所有分支的输出在通道维度上拼接，形成融合后的多尺度特征表示。LSTM 编码对多尺度卷积层提取的特征进行时序建模，捕捉长时依赖关系。通过注意力机制，强调对预测最为关键的时间步的影响，提高模型对重要时刻的响应能力。将注意力机制生成的上下文向量映射到预测目标的维度，实现多步预测。

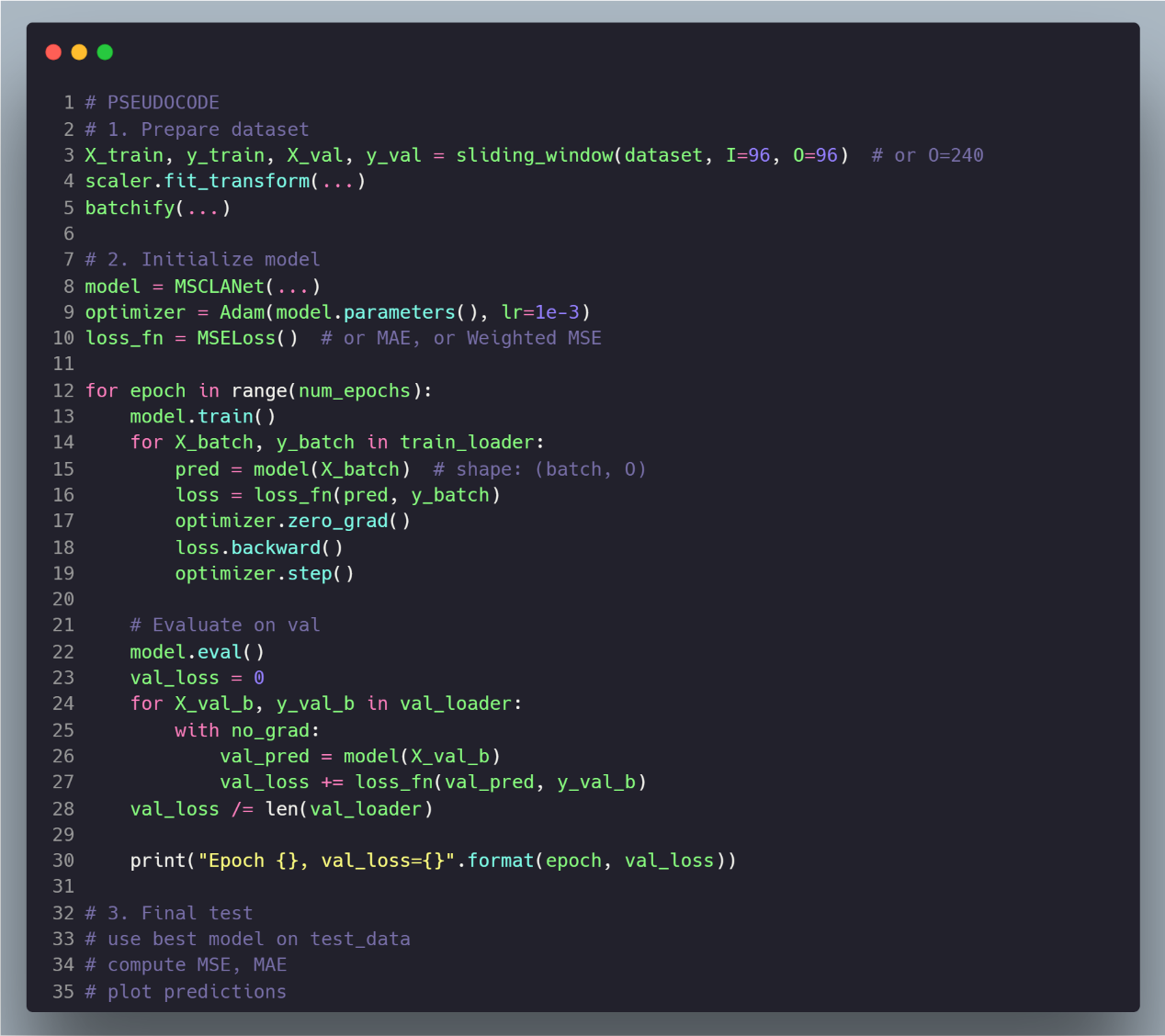
此外我观察上面的LSTM和Transformer，发现他们在预测峰值时，通常会有点保守，原因可能时这样：“峰值”出现频次往往较低，而且可能与常规数据分布差异明显。模型在最小化整体Loss时，更关注数据中占多数的中等或低值，而对高值错误并不特别敏感，于是模型会趋向于在峰值处更保守，因为大部分时间的预测做好了就能显著降低平均误差。于是我进一步调整了Loss函数，采用加权的Loss，针对超过一定阈值的值赋予更多的权重。

### 2.3.2 MSCLA 结构

MSCLA结构如下：

1. **输入层形状：**(batch,seq\_len=I,input\_size=9)，其中9表示特征列数（包括温度、湿度、天气、节假日、工作日、windspeed、cnt等）。
2. **多尺度卷积层（Multi-Scale CNN）：**若干并行卷积分支，例如 dilation = 1, 2, 4；每个分支输出维度cnn\_out\_channels；将并行分支输出在通道维度concat，得到 (batch,I,cnn\_out\_dim)。
3. **LSTM层：**隐藏单元数（hidden\_size）= 64，能处理长度I=96的序列；输出序列形状为 (batch,I,64)，每个时间步都有一个隐藏状态。
4. **注意力机制（Attention）**以一个可学习的查询向量或多头注意力，对所有时间步的隐藏状态做加权聚合；输出一个上下文向量 (batch,64)。
5. **全连接层（FC）：**将上述上下文向量映射到output\_size=96或output\_size=240，一次性预测对应时长的租借量曲线。

### 2.3.3 伪代码



### 2.3.4 训练与评估过程

1. **数据预处理：**构建滑动窗口；使用MinMaxScaler将temp, atemp, hum, windspeed, cnt等特征值映射到 [0,1] 区间。
2. **批量训练：**DataLoader：将(X,y)分批次输入模型，其中X为(batch,96,9)；采用合适损失函数，使用 Adam 优化器进行反向传播更新网络参数。
3. **验证集性能：**每个epoch结束后，在验证集上做前向预测，计算MSE、MAE并观察曲线，选择合适Epoch。
4. **最终测试：**使用在验证集上性能较好的模型，在test\_data上执行相同的滑动窗口预测；计算其MSE、MAE，并可视化预测曲线。
5. **多次训练和测试：**由于深度学习模型在不同随机初始化下可能存在一定波动，可以多次训练并统计均值与标准差；

# 结果与分析

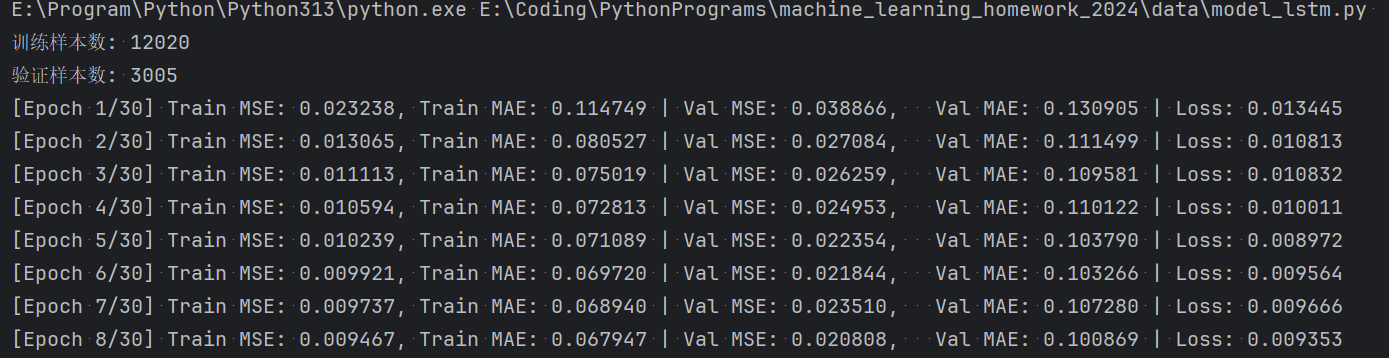
## 3.1 LSTM

### 3.1.1 训练过程

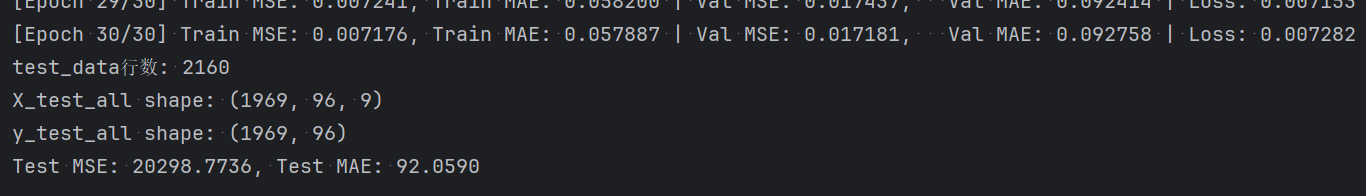
**首先是96小时**

通过观察验证集上的预测性能，差不多30左右指标不降反升，所以最后选择30的Epoch

训练过程的输出结果：



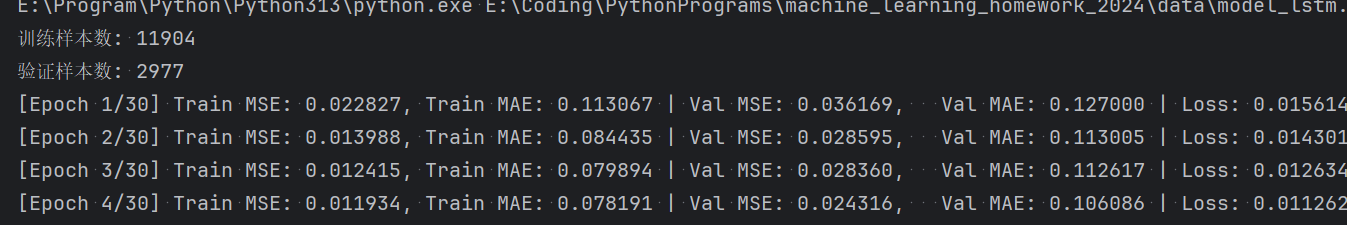
最后输出本次的结果和一段预测曲线（曲线见下一小节）



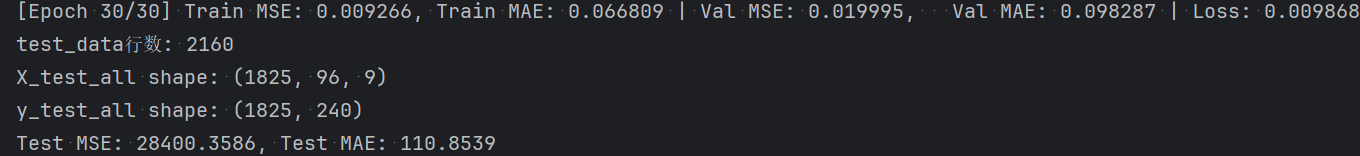
**然后是240小时**

还是选择30个Epoch

训练过程的输出结果：

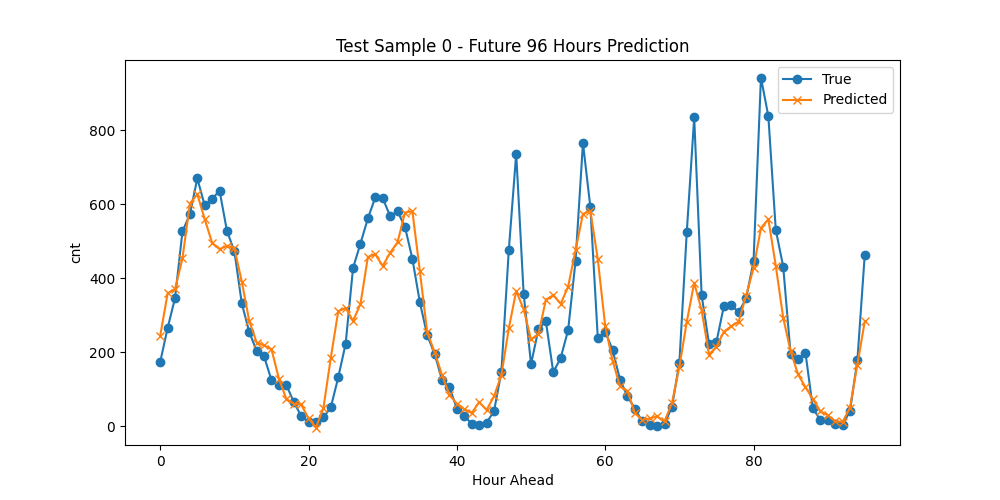


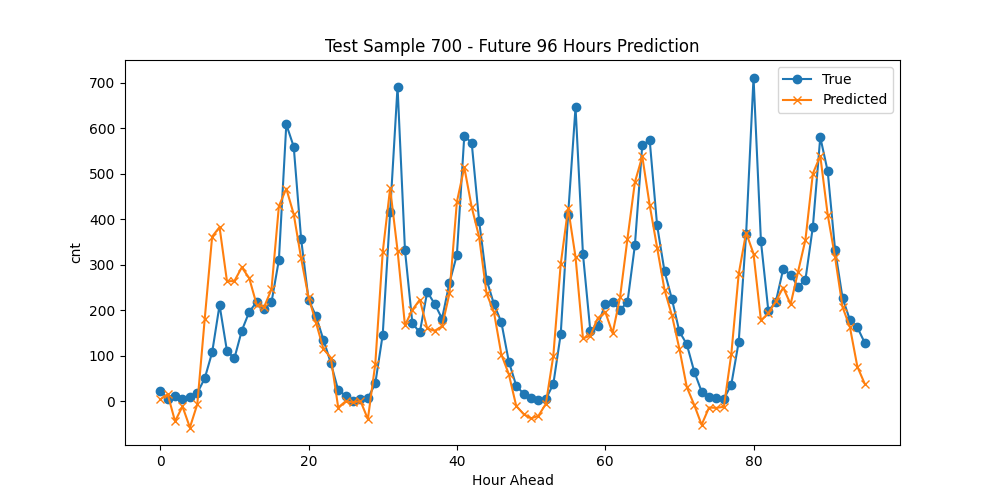
最后输出本次的结果和一段预测曲线



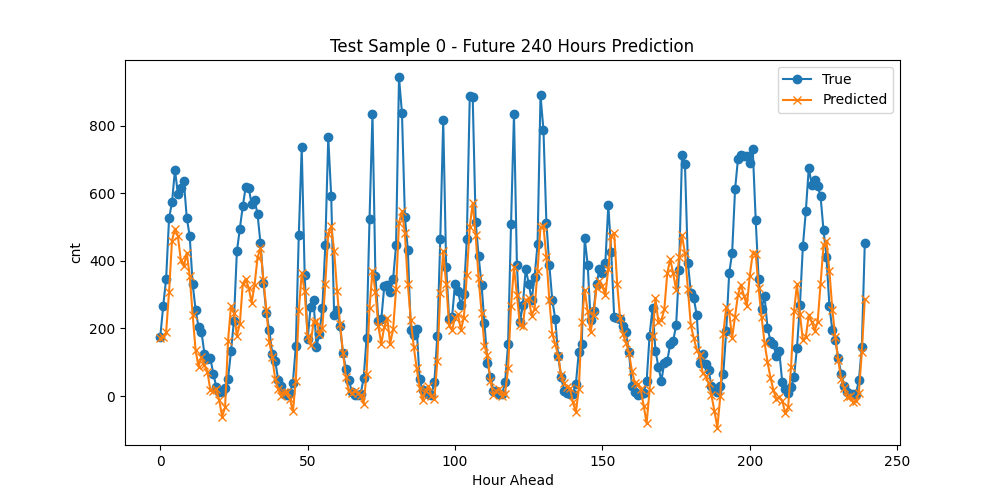
### 3.1.2 预测结果曲线

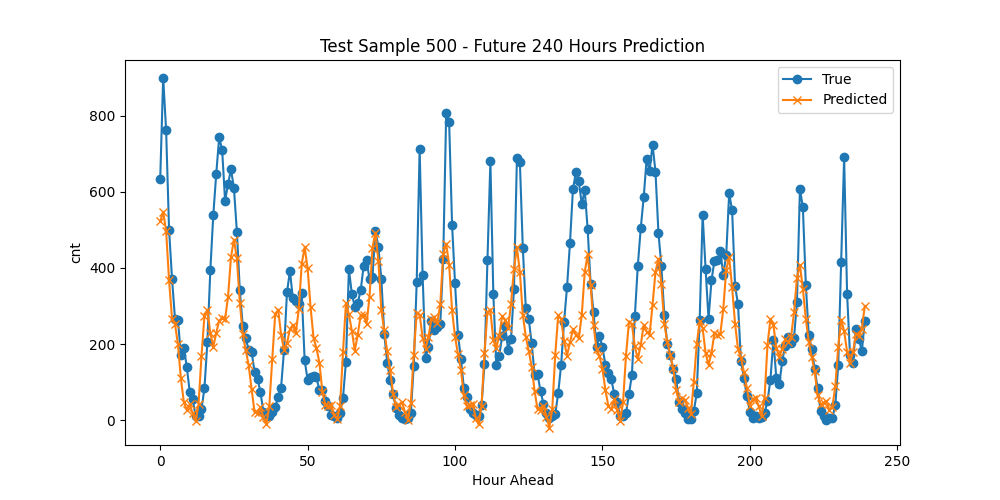
预测96小时，随机选取了其中两个窗口的预测结果绘制了如下的曲线





下面是预测240小时选择的两个窗口





### 3.1.3 模型稳定性

预测96小时的五轮的训练MSE、MAE以及标准差如下：

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 20298.7736 | 92.0590 |
| 2 | 21957.5224 | 96.4458 |
| 3 | 20447.1815 | 93.1595 |
| 4 | 20642.4489 | 92.3302 |
| 5 | 19893.1344 | 91.7428 |
| 标准差std | 699.5696930050955 | 1.7149297660254212 |

预测240小时的五轮的训练MSE、MAE以及标准差如下：

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 28400.3586 | 110.8539 |
| 2 | 29443.7118 | 113.2525 |
| 3 | 28447.2535 | 112.0095 |
| 4 | 28089.4154 | 110.2895 |
| 5 | 28593.1775 | 112.3442 |
| 标准差std | 455.11424871441073 | 1.059620268586817 |

### 3.1.4 分析

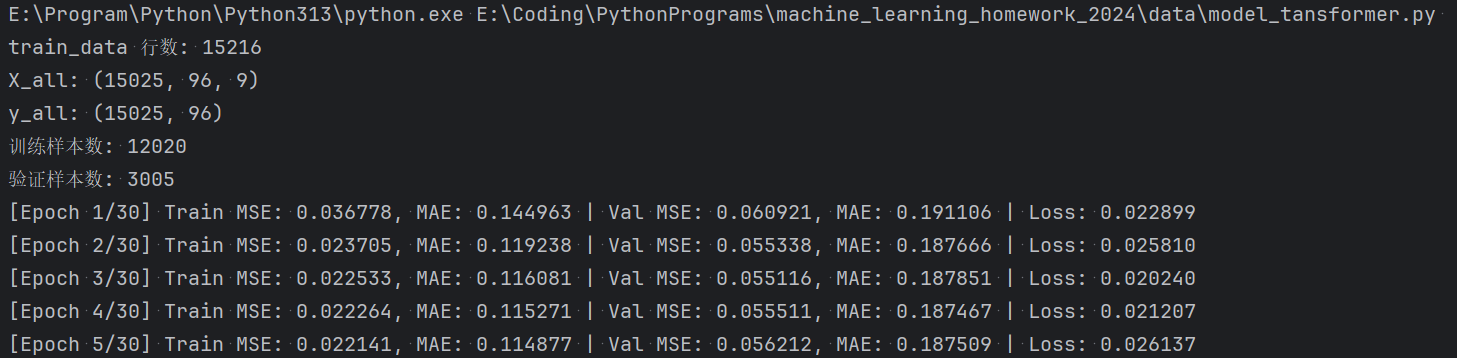
从96小时到240小时，MSE 和 MAE 都显著增加，说明预测范围越长，误差累积越明显。可能是LSTM在长序列预测中记忆能力有限，使得预测准确性下降。尽管240小时的预测误差更大，但标准差占均值的比例更低，表明240小时预测结果更稳定。总体上看LSTM能完成这个预测任务。96小时预测误差较低但标准差相对稍高，适合短期预测任务。240小时预测误差较高但标准差较小，预测效果更稳定但需改善长期预测性能。

## 3.2 Transformer

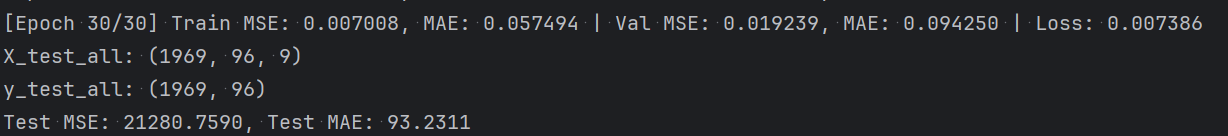
### 3.2.1 训练过程

预测96小时还是选择用30个Epoch

训练过程的输出结果：

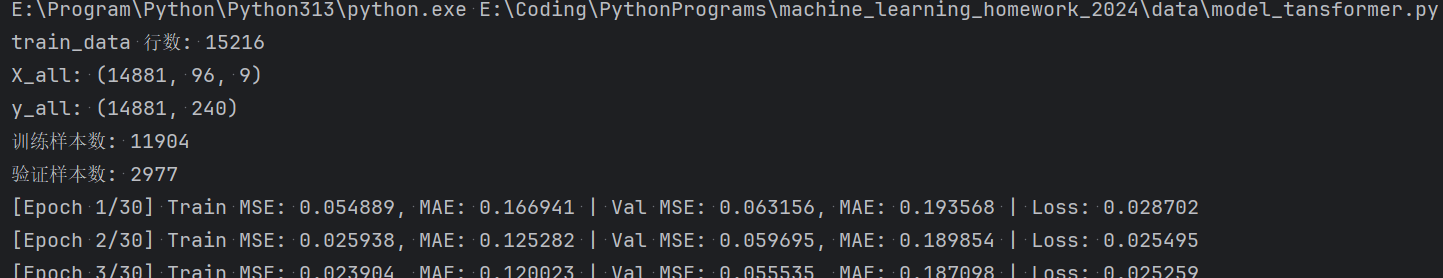


最后输出本次的结果和一段预测曲线（曲线见下一小节）

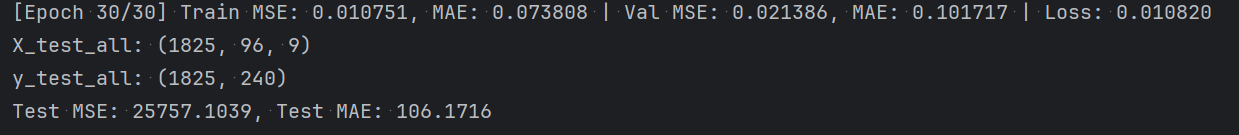


预测240小时

训练过程的过程：

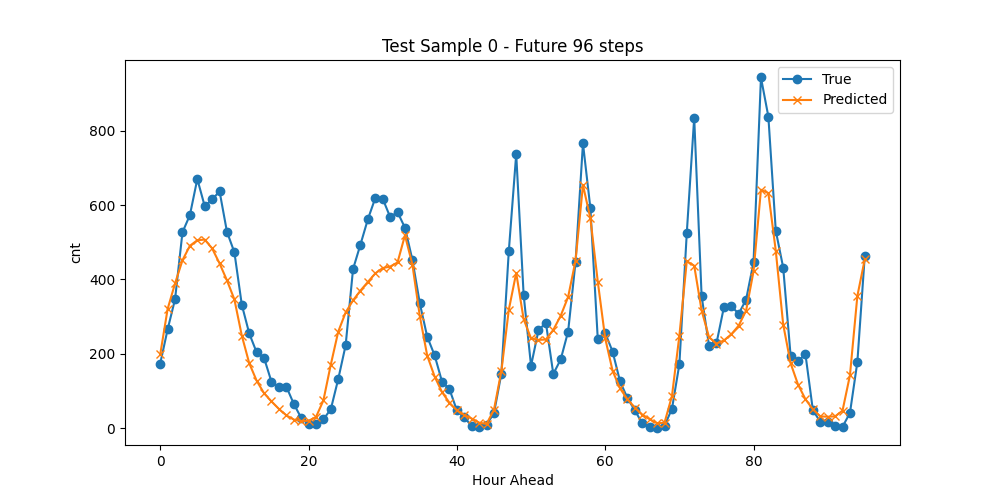


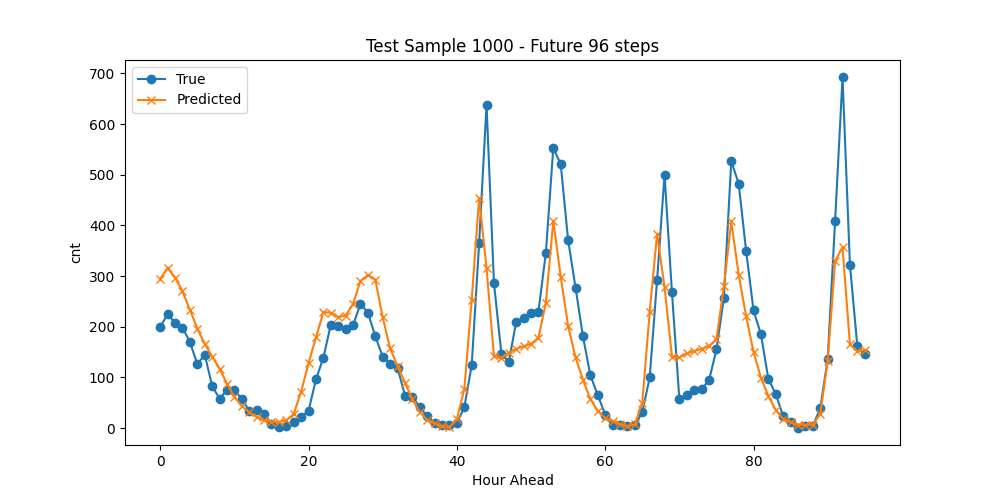
结果：



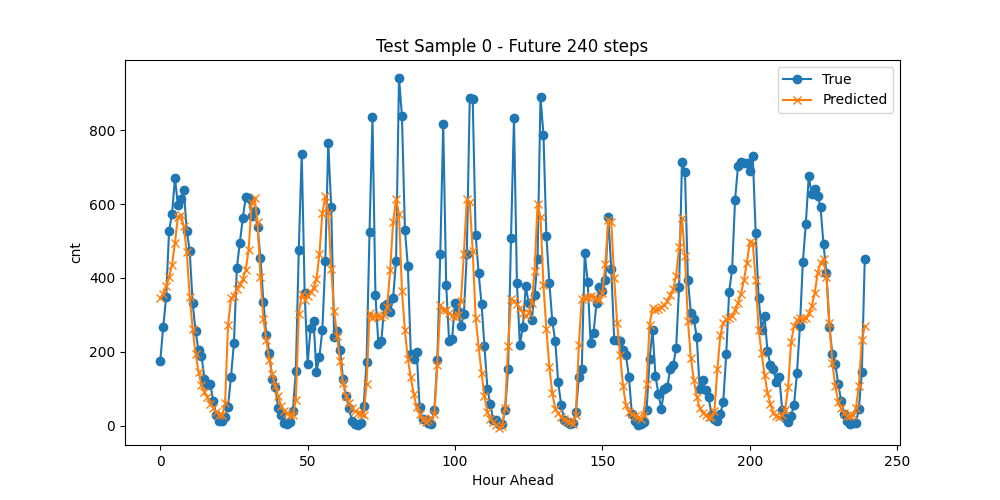
### 3.2.2 预测结果

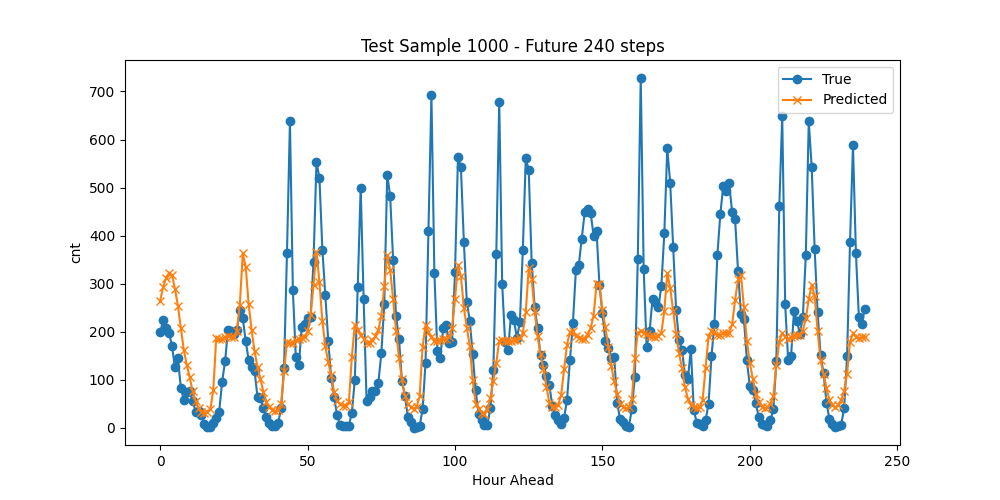
预测96小时，随机选取了其中两个窗口的预测结果绘制了如下的曲线





预测240小时，随机选取了其中两个窗口的预测结果绘制了如下的曲线





### 3.2.3 模型稳定性

96小时预测

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 20546.1682 | 92.2503 |
| 2 | 22781.6179 | 96.7133 |
| 3 | 21280.7590 | 93.2311 |
| 4 | 20616.1581 | 91.7433 |
| 5 | 20516.5902 | 91.9139 |
| 标准差std | 863.720755606532 | 1.8448748232874785 |

240小时预测

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 25757.1039 | 106.1716 |
| 2 | 28628.3153 | 111.7106 |
| 3 | 26425.2853 | 106.2438 |
| 4 | 27164.8546 | 109.4246 |
| 5 | 28750.3871 | 112.7591 |
| 标准差std | 1185.052943959389 | 2.717053275591043 |

### 3.2.4 分析

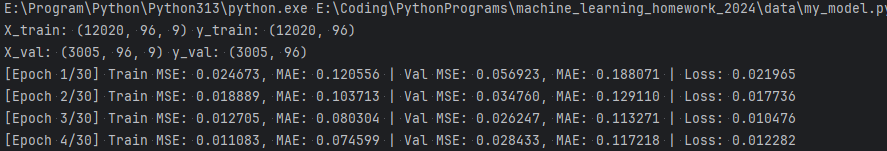
从96小时到240小时，MSE 和 MAE 都显著增加，说明预测范围越长，在Transformer中的误差累积也是越来越明显。LSTM 更适合稳定性要求高、短时间范围内的预测任务。Transformer 更适合捕捉长期依赖关系的复杂任务，但需要优化以提高训练结果的稳定性。Transformer的240小时MSE和MAE明显低于LSTM，表明其在长期预测中的效果更好。然而，Transformer 的标准差较大，说明在多次训练中，Transformer 的表现容易受到随机初始化、超参数选择的影响，模型的表现波动较大。

## 3.3 MSCLA

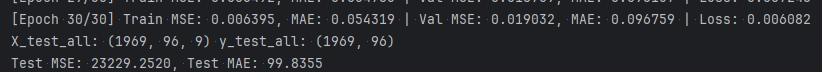
### 3.3.1 训练过程

预测96小时还是选择用30个Epoch

训练过程的输出结果：

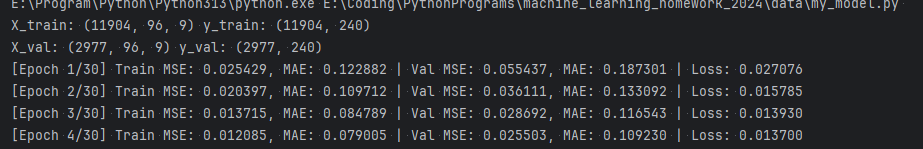


最后输出本次的结果和一段预测曲线

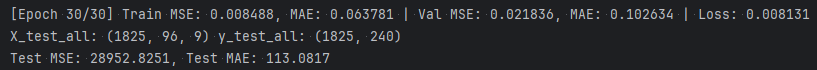


预测240小时

训练过程的过程：

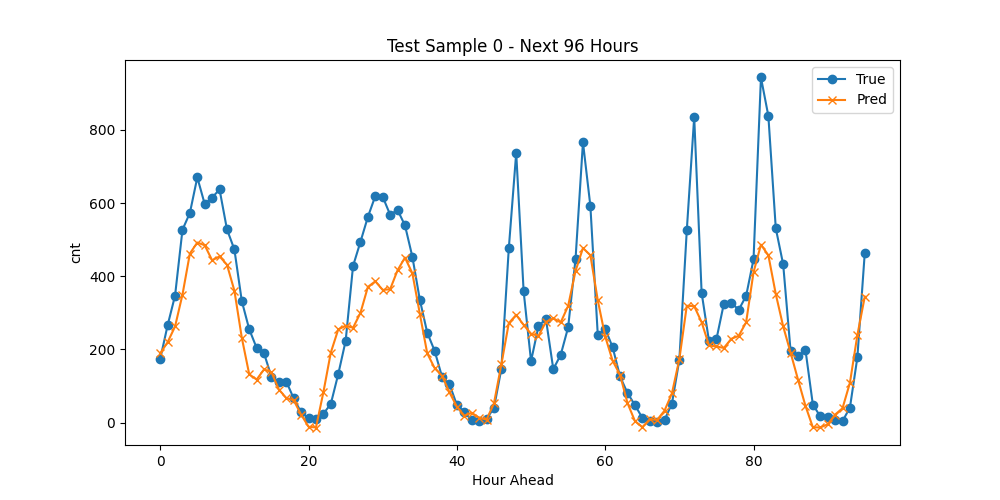


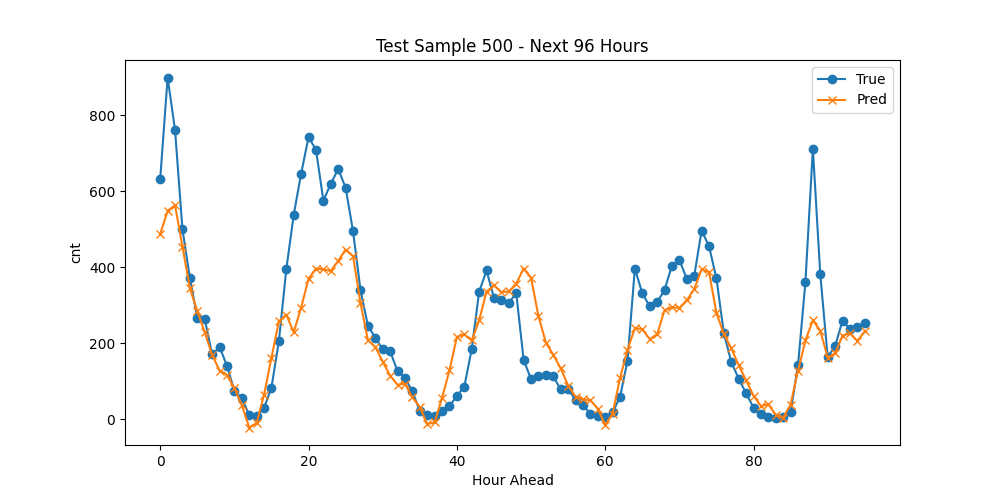
结果：



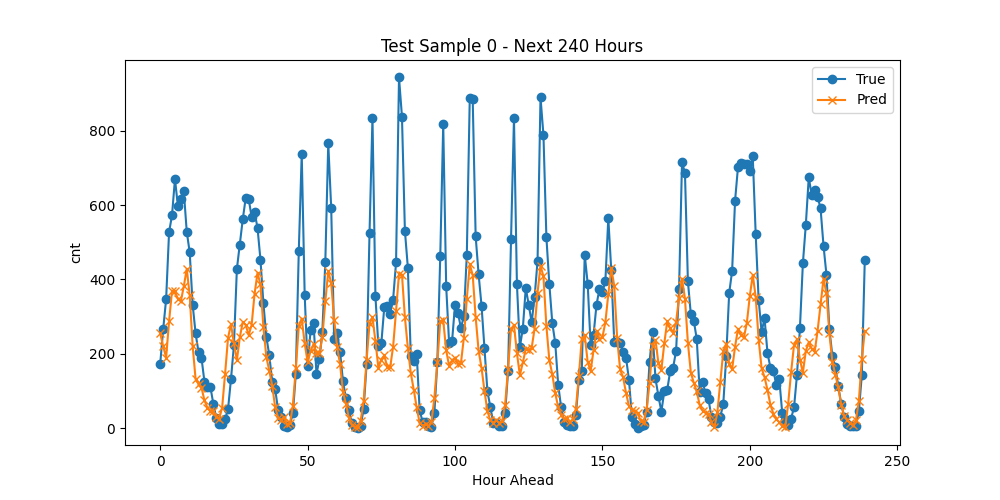
### 3.3.2 预测结果

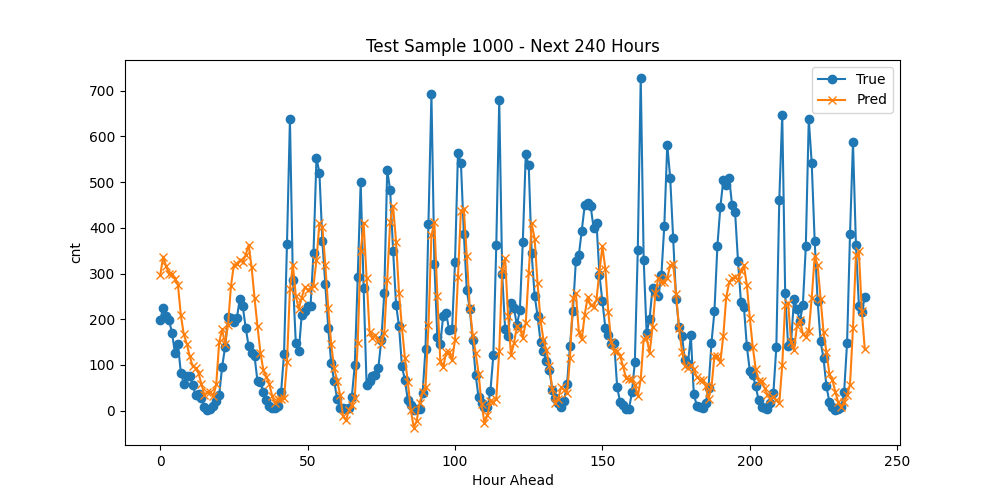
预测96小时，随机选取了其中两个窗口的预测结果绘制了如下的曲线





预测240小时，随机选取了其中两个窗口的预测结果绘制了如下的曲线





### 3.3.3 模型稳定性

96小时预测

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 22741.1016 | 97.9898 |
| 2 | 22867.0336 | 97.6864 |
| 3 | 26185.6055 | 108.4054 |
| 4 | 25253.4535 | 104.0220 |
| 5 | 25531.4830 | 107.7608 |
| 标准差std | 1430.522987041666 | 4.606844067862509 |

240小时预测

|  |  |  |
| --- | --- | --- |
| 轮次 | MSE | MAE |
| 1 | 30550.4088 | 115.6367 |
| 2 | 28941.9955 | 113.9501 |
| 3 | 31117.4247 | 119.7808 |
| 4 | 29221.3833 | 114.0961 |
| 5 | 30351.8528 | 116.4369 |
| 标准差std | 823.9299875022033 | 2.118597002169121 |

### 3.3.4 分析

从上面可以看出，新模型其实并没有表现出非常好的效果，可能是中间参数以及Epoch数量还有优化空间，此外从图中可以产出新加入的加权Loss应该是起到了一定的作用，高点相比较LSTM和Transformer可能好了一点，但是也不明显，可能还是要更加细致的方案，而直接超过阈值就增加权重，可能并不太合适。模型的稳定性正常相比LSTM和Transformer都区别不大。存在的问题还有中间Epoches的选择，有时选择小，模型不充分，选择大了容易过拟合效果反而下降，但是模型训练过程又有一定的随机性，所以大小有时也不太能通过训练时输出的指标精确判断，只有大致的范围。

# 讨论和总结

本次实验完成了LSTM和Transformer的时间序列预测任务。

LSTM能够有效地捕捉长时间序列中的长期依赖关系。它通过引入记忆单元和门控机制（输入门、遗忘门、输出门）来解决传统RNN在处理长序列时遇到的梯度消失问题。它的记忆单元能够在每个时间步骤中传递重要的信息，同时遗忘无关的部分，从而使网络能够在较长的时间跨度上进行有效学习。

Transformer是一种基于自注意力机制的深度学习模型，广泛应用于自然语言处理任务。Transformer完全摆脱了顺序计算的限制，采用了自注意力机制来并行处理序列中的所有元素，显著提高了计算效率。它通过多头注意力机制对输入的每个位置进行加权平均，捕捉序列中各个部分之间的关系。它具有并行处理能力并且对长范围依赖的建模能力更好。

在共享单车租赁数量预测的这个问题中，租赁数量受多种因素的影响，包括天气、季节、工作日非工作日、节假日等，而这些因素随时间变化是典型的时间序列预测问题。LSTM在处理周期性和季节性数据时表现较好，特别是通过滑动窗口方式构建的序列数据。而Transformer在捕捉周期性和季节性变化方面也很适用，注意力机制可以帮助模型集中注意力于过去的关键时间点。这两个都能很好处理此类预测问题。

通过96小时和240小时这两个的预测结果可以看出LSTM在短期预测中能够依赖相对较近的历史数据进行训练，捕捉短期的趋势变化。其记忆单元能够有效存储并处理最近时间段的数据，帮助模型在短期内做出较准确的预测。而Transformer模型在处理更长时间序列时，能够通过并行处理并捕捉时间序列中较远距离的依赖关系，因此适合用于较长时间跨度的预测。其自注意力机制能够处理长时间依赖，并且在长时间预测任务中，能够减少传统 LSTM在长期依赖中可能遇到的梯度消失问题。

在epoch的选择上如果epoch设置过小，模型可能无法充分学习到数据的规律，导致欠拟合。如果epoch设置过大，模型可能会开始对训练数据过度拟合，导致过拟合，此时训练误差继续降低，但验证误差开始上升。可以通过监控训练误差和验证误差的变化来判断。当验证误差开始上升时，表示模型可能已经学习到了训练集的噪声，开始出现过拟合，此时就应该停止训练。还可以使用早停，允许验证误差在若干轮训练内没有显著改善后才停止，可以在每一轮训练中保存验证误差最小的模型，还可以交叉验证在不同的数据集划分上训练和评估模型的泛化能力。但是本次实验没有使用特殊的方法，直接通过观察训练过程中的验证误差和损失函数的变化趋势，确定的Epoch大致为30，当然这种方法并不准确，可能会由于模型随机性不是很稳定。

提出的新模型多尺度卷积 + LSTM + 注意力的MSCLA模型在这个预测任务中也完成了任务。通过结合多尺度卷积提取不同时间尺度的局部特征、LSTM捕捉长时依赖关系以及注意力机制强调关键时间步的影响，MSCLA模型在捕捉复杂时序模式和应对峰值预测方面展现出一定的性能。此外，还通过自定义加权损失函数的策略，进一步提升了模型对高值区域的预测能力。遗憾的是实际效果可能跟其他模型拉不开差距，可能还需要进行进一步模型调优。

最后再回到这个预测任务上，通过查看数据集，我觉得还是有很多信息可以挖掘的，目前模型以及能准确反映出同一天内的波动情况，正确描述一天内的变化趋势，而对于周、月、年来说是否也隐含一些关系；还有题中给出的casual和registered两类租赁分开，是否他们两个对于某些日期也有不一样的关系，或者他们之间是否也有一定的比例联系，这些关系可能都可以拿出来单独分析。

【参考文献】

[1] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[2] Vaswani A. Attention is all you need[J]. Advances in Neural Information Processing Systems, 2017.

[3] Bai S, Kolter J Z, Koltun V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling[J]. arXiv preprint arXiv:1803.01271, 2018.

[4] Van Den Oord A, Dieleman S, Zen H, et al. Wavenet: A generative model for raw audio[J]. arXiv preprint arXiv:1609.03499, 2016, 12.

【代码】

见GitHub

[Blank-Sky/machine\_learning\_homework\_2024](https://github.com/Blank-Sky/machine_learning_homework_2024)