基于传统机器学习、LSTM 和 Transformer 的血糖水平时序预测

一、引言

糖尿病是一种影响全球数亿人群体健康的慢性疾病。血糖水平的动态监测和精确预测对于糖尿病患者的日常健康管理至关重要，能有效预防急性并发症和长期并发症。近年来，随着连续血糖监测（CGM）技术的发展，获取详细的血糖时间序列数据成为可能，为基于机器学习的血糖预测模型提供了基础。本研究尝试使用三种不同的模型——支持向量机(SVM)及神经网络(包括长短期记忆循环神经网络 LSTM 和时间序列 Transformer 模型)，通过融合患者的静态特征和动态特征，区分处理数值特征和分类特征，对糖尿病患者的血糖水平进行预测，初步分析了模型的预测结果，并探讨了可能的改进方向。

二、领域知识背景下的数据预处理

本研究使用的原始数据出自“Chinese diabetes datasets for data-driven machine learning”，其中包括来自上海地区 1 型糖尿病（T1DM）和 2 型糖尿病（T2DM）患者的实时血糖 CGM 动态数据以及患者的静态特征信息。CGM 血糖数据以 15 分钟为间隔记录，并包含一些与血糖波动相关的动态特征。静态特征数据则来自于患者的健康记录和生活习惯调查。

数据集概况：
Shanghai_T1DM_Summary：记录了 12 名患者的基本静态信息。
Shanghai_T2DM_Summary：记录了 100 名患者的基本静态信息。
T1DM 文件夹：共 16 条时间序列记录。在 12 人中，10 人记录 1 次，2 人记录 3 次。
T2DM 文件夹：共 109 条时间序列记录。在 100 人中，92 人记录 1 次，7 人记录 2 次，1 人记录 3 次。

在进行血糖预测模型的构建之前，首先需要深入了解血糖调节机制和影响血糖水平的因素。血糖水平的是受到多种因素的综合影响的，包括饮食摄入、胰岛素使用、非胰岛素降糖药物、运动、应激状态等。我们需要在数据预处理中提取和构建关键特征。在此处，我们区分了动态特征和静态特征。

动态特征是指会随时间变化的特征。这些特征主要包含于 Shanghai_T1DM 中的 16 份 excel 文件和 Shanghai_T2DM 中的 109 份 excel 文件。这些文件中记录了我们需要预测的时间序列数据，记录了连续血糖监测（CGM）数据、毛细血糖（CBG）水平、血酮体水平、饮食摄入、胰岛素的摄入剂量和非胰岛素降糖药摄入等等。

对于饮食而言，碳水化合物的摄入是血糖水平波动的主要因素之一。我们曾设想过计算每餐摄入具体卡路里的值，但在此处批量实现较为困难，所以我们直接添加 0-1 变量 take_food 用以判断患者是否在该时段进食。由于数据存在的完整性问题，在处理每个文件时会检查“Dietary intake”和“饮食”两列，如果这两列中的任一列包含了数据，则 take_food 列将标记为 1，否则标记为 0。处理完成后，“Dietary intake”和“饮食”两列将被移除，以减少数据的冗余和混乱。

根据数据我们发现，对于糖尿病患者来说，通常在其饮食前都会服用相关降糖药，所以在饮食后，大多数情况下其血糖并不会显著升高，甚至有可能降低。所以，胰岛素类药物和非胰岛素类降糖药物的使用对血糖水平均有显著影响。我们需了解药物的种类、剂量

和时间信息。可是在数据中，存在着多种不同的药物名称，且不同患者的耐药性不同，我们很难量化药物对患者的影响，在这里，我们仍然使用了简化的思想，添加与原始列名相同的 0-1 变量列，用以判断在该时间点患者是否服用相关药物。

静态特征是指在一次时间序列中不会随时间发生变化的特征。在本研究中，静态特征包括了所有基本的个人信息和健康状态，这些都是在病人的整个观察期内不会改变的。在 Summary 摘要表格中，诸如性别、年龄、身高、体重、BMI、吸烟饮酒史、糖尿病类型、糖尿病持续时间、糖尿病并发症、合并症以及低血糖发生情况等等都是静态特征。

糖尿病分为 1 型糖尿病（T1DM）和 2 型糖尿病（T2DM）。在发病原因方面，T1DM 主要是由于胰岛素分泌减少或消失，这是由于胰岛 β 细胞受到自身免疫系统的破坏所致。而 T2DM 的发病原因则与胰岛素抵抗（细胞对胰岛素反应减弱）及胰岛素分泌减少有关。据研究表明，T1DM 相较于 T2DM，血糖波动更大，发生高血糖、低血糖的风险更高。所以在此处做区分。

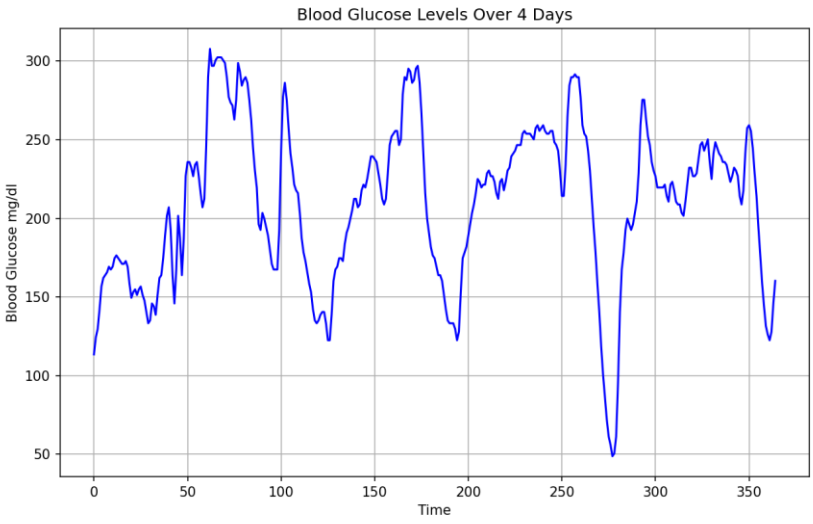
从现代医学角度看，抽烟和饮酒史均会导致更高的高血糖风险；高血糖会增加糖尿病酮症酸中毒的风险；糖尿病合并低血糖症是指因为治疗不当而致的血糖持续性过低的现象，会导致血糖更大的波动。所以除了以年份统计的抽烟史我们将其作为数值特征，其余变量我们都将加入分类特征。

最后，我们剔除了一些与当前血糖水平关联较小的特征，其余有关特征我们均保持原样，作为数值特征。然后我们进行了缺失值和异常值的处理，进一步将数据规范化，确保所有数据都是干净且格式一致的。（在最后的附录中会提到）

三、 特征工程和特征重要性

3.1 特征工程

我们采用了与时间序列数据相关的特征工程方法，提取与时间相关的特征，并进行适当的编码。



首先我们需要确定数据集中是否存在明显的时间戳。如果数据具有明显的周期性（如日周期、周周期等），可以添加如小时、星期几等特征。从图中可知，人体血糖水平呈现较为明显的日周期性特征。所以添加 hour（即当天所处的小时数），作为特征变量。

同时，我们创建了 CGM_lag1 和 CGM_lag2 作为滞后特征，它们代表的是前 1 个和 2 个时间点的数据，这对于捕捉时间序列的动态作用较大。

对于分类特征，我们曾考虑使用独热编码，即使用 N 位状态寄存器来对 N 个状态进行编码，每个状态都由他独立的寄存器位，并且在任意时候，其中只有一位有效。这样，数据就会变成稀疏的。但是它存在的缺点是，当类别的数量很多时，特征空间会变得非常大。在这种情况下，一般可以用 PCA（主成分分析）来减少维度，但这未必能保留原始时间序列数据中的时间依赖特征。所以放弃使用独热编码。

处理步骤：

- 加载患者的静态和动态特征，将静态特征合并于包含 CGM 的时间序列数据中。
- 处理动态分类特征，将其转换成 0-1 整数编码。
- 处理静态分类特征，将其转换成 0-1 整数编码。
- 移除一些与血糖预测关联较少的列。
- 添加时间相关特征和滞后特征，以捕捉时间序列信息。
- 移除所有空列和非数值列。

最终，我们形成了以下特征作为模型输入：

分类特征 (Categorical Features)

- Insulin dose - s.c.: 皮下胰岛素剂量
- Non-insulin hypoglycemic agents: 非胰岛素降糖药
- CSII - bolus insulin (Novolin R, IU): 胰岛素泵-餐前胰岛素
- Insulin dose - i.v.: 静脉注射胰岛素剂量
- take_food: 进食标记
- Gender (Female=0, Male=1)：性别
- Alcohol Drinking History (drinker/non-drinker)：是否有饮酒史
- Type of Diabetes：糖尿病种类
- Acute Diabetic Complications：是否有酮症酸中毒
- Hypoglycemia (yes/no)：是否有低血糖症

数值特征 (Numerical Features)

- CGM (mg / dl): 连续血糖监测数据
- Age (years)：年龄
- Height (m)：身高
- Weight (kg)：体重
- BMI (kg/m2)：BMI 指数
- Smoking History (pack year)：抽烟年份
- Duration of Diabetes (years)：糖尿病年份
- Hour：小时数
- CGM_lag1：滞后特征 1
- CGM_lag2：滞后特征 2

四、模型的选择与构建

4.1 SVM

4.1.1 原理

支持向量机 (SVM) 是一种监督学习算法，可用于解决分类和回归问题。在本研究中，我们使用 SVM 进行回归预测，即支持向量回归 (SVR)。

SVR 的基本原理是寻找一个最优的超平面，使得训练数据集中所有样本点到该超平面的距离最小，同时允许一定程度的误差容忍。

与传统的线性回归不同，SVR 通过引入核函数将数据映射到高维空间，从而能够捕捉数据中的非线性关系。

SVR 的目标函数包含两个部分：

1. 最小化误差: 最小化所有样本点到超平面的距离，即预测值与真实值之间的差异，通常使用 ε-不敏感损失函数来度量。
2. 最大化间隔: 最大化超平面两侧的间隔，以提高模型的泛化能力

核函数的选择对于 SVR 的性能至关重要。常用的核函数包括：

1. 线性核: 适用于线性可分的数据集。
2. 多项式核: 适用于非线性可分的数据集，可以捕捉特征之间的多项式关系。
3. 径向基函数核 (RBF): 一种常用的非线性核函数，可以捕捉特征之间的复杂关系。
4. Sigmoid 核: 类似于神经网络中的激活函数，可以捕捉特征之间的非线性关系。

4.1.2 模型定义

创建 SVR 模型，并使用 MultiOutputRegressor 将其封装成多输出回归器，用于同时预测多个时间步的血糖值。

```
5
6  # 创建SVM回归模型
7  svm = SVR()
8
9  # 创建多输出回归器
10 multi_output_svr = MultiOutputRegressor(svm)
```

4.1.3 模型参数调优及评估

参数调优: 使用 GridSearchCV 对模型进行参数调优，寻找最佳的核函数、正则化系数 C 和容忍误差 epsilon。

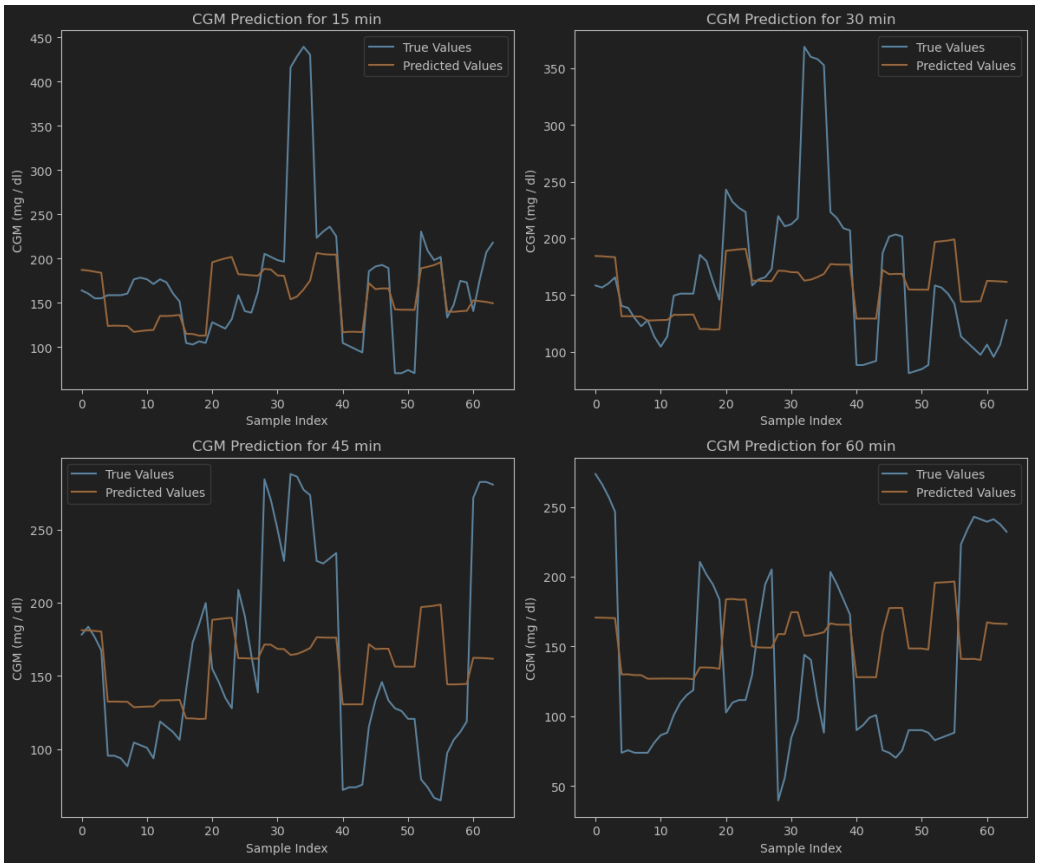
```python
# 定义参数网格
param_grid = {
    'regressor__estimator__kernel': ['linear', 'poly', 'rbf', 'sigmoid'],
    'regressor__estimator__C': [0.1, 1, 10, 100],
    'regressor__estimator__epsilon': [0.1, 0.2, 0.5, 1]
}

# 使用GridSearchCV进行参数调优
grid_search = GridSearchCV(pipeline, param_grid, cv=3,
 scoring='neg_mean_absolute_error', verbose=2, n_jobs=-1)
grid_search.fit(X_train, y_train)
```

模型评估: 使用测试集评估模型性能，计算平均绝对误差（MAE）

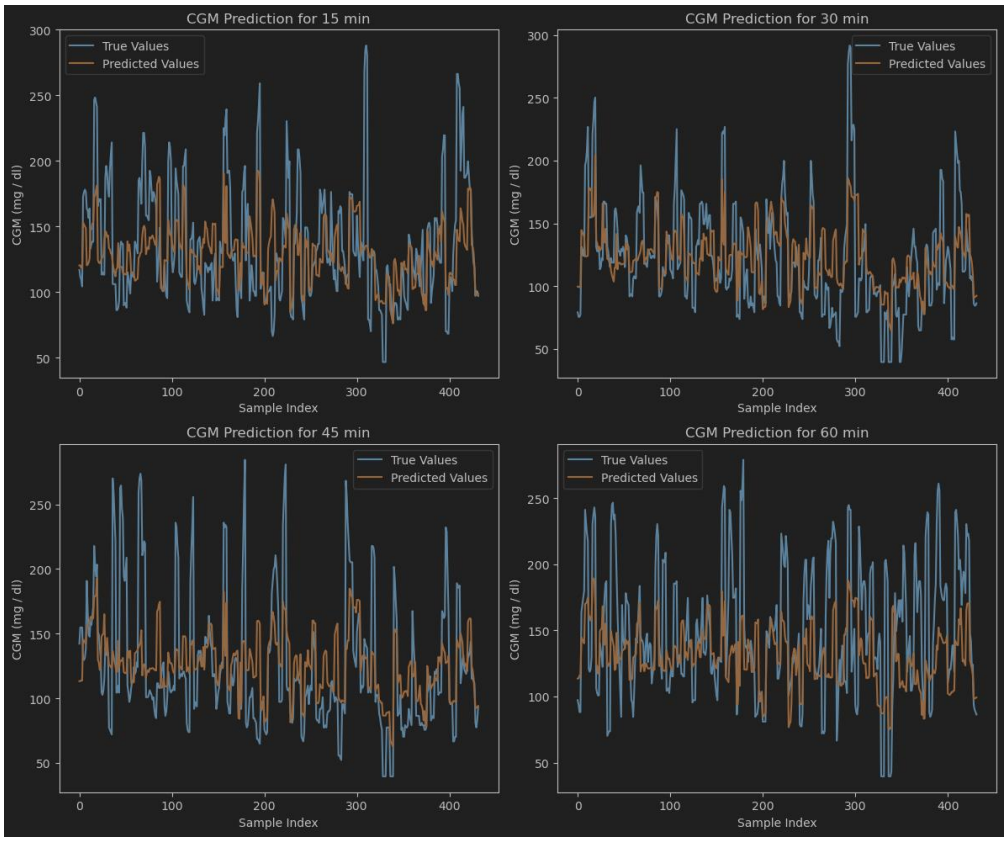
```python
# 使用最佳参数进行预测
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)

# 评估模型性能
mae = mean_absolute_error(y_test, y_pred, multioutput='raw_values')
print("Mean Absolute Error for each time interval: ", mae)
```

4.1.4 分析与可视化

对患者整体进行预测，得结果如下:



T1DM 患者四个时间节点的 MAE 分别为：47.36455181、44.38376449、55.29712483、62.66664197。



T2DM 患者四个时间节点的 MAE 为：28.37003607、25.05599569、30.27658962、36.70755367。

再对两类病人中的每个病人进行单独预测，得到：

```
Mean Absolute Error for each time interval for patient processed_1001:
[41.76173769 41.28046329 26.84803467 61.40804289]
Mean Squared Error for each time interval for patient processed_1001:
[1757.30044784 1714.32054669  756.64975699 3872.52895227]
Mean Absolute Error for each time interval for patient processed_1002:
[38.69231821 27.10923026  9.24467902  7.79879358]
Mean Squared Error for each time interval for patient processed_1002:
[1607.82285789  735.44346538  108.59810579  107.39294541]
Mean Absolute Error for each time interval for patient processed_1003:
[27.86458417 28.63960929 35.41081307 66.04225493]
Mean Squared Error for each time interval for patient processed_1003:
778.8187791  1053.81054471 1739.67176053 4461.27965123]
Mean Absolute Error for each time interval for patient processed_1004:
[51.06455207 58.56260993 34.97952697 65.98129416]
Mean Squared Error for each time interval for patient processed_1004:
[2623.71775002 3473.09178333 1315.90563378 4372.35232436]
Mean Absolute Error for each time interval for patient processed_1005:
[33.02378878  9.91726617 24.3830058  25.8356227 ]
```

T1DM 患者部分预测 MAE

该模型在 T1DM 患者预测上的平均 MAE 为：2,662.182582525

```
Mean Absolute Error for each time interval for patient processed_2094:
[117.64075068  70.03739109  36.99128372  89.3443712 ]
Mean Squared Error for each time interval for patient processed_2094:
[13854.33587742  4974.21287386  1370.65356739  8036.98284013]
Mean Absolute Error for each time interval for patient processed_2095:
[17.85449041 38.92510036 95.32895475 29.6793525 ]
Mean Squared Error for each time interval for patient processed_2095:
365.87034415 1679.11126311 9162.06241134 1047.85239346]
Mean Absolute Error for each time interval for patient processed_2096:
[49.38947479 25.16307934 16.72733841 44.40578472]
Mean Squared Error for each time interval for patient processed_2096:
[2444.84506358  634.1290229   281.99962158 2025.19899884]
Mean Absolute Error for each time interval for patient processed_2097:
[11.97915781 19.615847   13.42744391 66.84668123]
```

T2DM 患者部分预测 MAE

该模型在 T2DM 患者预测的平均 MAE 为：1,615.44574059

4.2 LSTM

4.2.1 原理

长短期记忆网络（LSTM）是一种特殊类型的循环神经网络（RNN），专门设计来解决传统 RNN 在处理长序列数据时遇到的梯度消失问题。该网络拥有 LSTM 单元，这些单元除了 RNN 的外部循环外，还具有内部循环（自环）。每个单元具有与普通循环网络相同的输入和输出，但具有更多的参数和一套门控单元系统。LSTM 的核心即是三个门控系统：输入门、遗忘门和输出门。这些门控制信息的保存、更新和输出，使网络能够在需要时保留或丢弃信息。相比于普通 RNN，LSTM 通过控制门能够维持长期依赖关系，有效地保存并利用长期的信息，使其能更好地学习长序列性质。

LSTM 的核心在于其独特的细胞结构和门控机制。LSTM 细胞包含三个门控单元：遗忘门、输入门和输出门，以及一个细胞状态。

细胞结构：

1. 细胞状态 (Cell State)：贯穿整个 LSTM 细胞，像一条信息传送带，用于存储和传递长期信息。
2. 遗忘门 (Forget Gate)：决定从细胞状态中丢弃哪些信息。它接收当前时间步的输入和前一时间步的隐藏状态，通过 sigmoid 函数输出一个 0 到 1 之间的数值，控制哪些信息被遗忘。
3. 输入门 (Input Gate)：决定将哪些新信息存储到细胞状态中。它也接收当前时间步的输入和前一时间步的隐藏状态，通过 sigmoid 函数确定哪些信息需要更新，并通过 tanh 函数生成新的候选状态。
4. 输出门 (Output Gate)：决定基于细胞状态输出哪些信息。它接收当前时间步的输入和前一时间步的隐藏状态，通过 sigmoid 函数决定输出哪些信息，并将细胞状态通过 tanh 函数处理后与输出门的结果相乘得到最终输出。

LSTM 工作流程：

1. 遗忘门根据当前输入和前一时间步的隐藏状态决定从细胞状态中丢弃哪些信息。
2. 输入门决定哪些新信息需要被存储到细胞状态中，并生成新的候选状态。
3. 将旧的细胞状态与遗忘门的输出相乘，实现信息的丢弃。
4. 将新的候选状态与输入门的输出相乘，并将结果与更新后的细胞状态相加，实现信息的更新。
5. 输出门根据更新后的细胞状态和当前输入决定最终的输出。

LSTM 非常适合血糖时间序列预测，主要基于以下几个原因：

1) 血糖水平并非独立存在，而是受到先前时间点血糖值、胰岛素注射、饮食和其他因素的综合影响。LSTM 擅长捕捉时间序列数据中的**长期依赖关系**，能够学习这些复杂关系并进行更准确的预测。
2) LSTM 的门控机制允许其**选择性地记忆和遗忘**信息。对于血糖预测，LSTM 可以学习记住患者过去的血糖模式、对治疗的反应以及其他重要因素，并在预测未来血糖水平时利用这些信息。
3) 血糖水平的变化通常呈现**非线性**模式。LSTM 作为一种非线性模型，能够有效地捕捉和建模这些非线性关系，从而提高预测精度。

4.2.2 模型定义

本研究采用 LSTM 模型进行血糖预测，并使用 Python 语言和 Keras 深度学习库实现。

LSTM 层数和单元数: 代码使用了两层 LSTM，每层包含 50 个神经元。

Dropout 层: 代码在 LSTM 层之间添加了 Dropout 层，有效防止过拟合。

```
# 创建LSTM模型
model = Sequential()
model.add(LSTM(units=50, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dropout(0.2))  # 添加Dropout层以防止过拟合
model.add(LSTM(units=50))
model.add(Dropout(0.2))  # 添加Dropout层以防止过拟合
model.add(Dense(1))
```

4.2.3 模型训练与验证

自定义指标: 代码定义了自定义的准确率指标 custom_accuracy，根据血糖预测值与真实值之间的差异是否小于阈值来计算准确率。

```
# 计算均绝对误差（MAE）和均方误差（MSE）
mae = mean_absolute_error(y_val, y_pred)
mse = mean_squared_error(y_val, y_pred)
acc = np.mean(np.abs(y_val - y_pred) < 0.1)  # 自定义准确度
```

损失函数: 代码使用均方误差（MSE）作为损失函数，用于度量模型预测值与实际值之间的差异。

优化器: 代码使用 Adam 优化器。

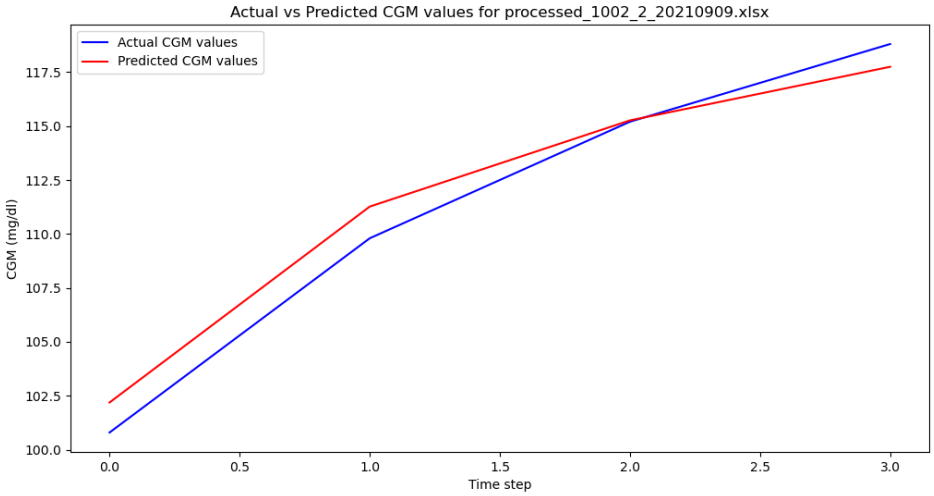
```
# 设置学习率
learning_rate = 0.001
optimizer = Adam(learning_rate=learning_rate)

model.compile(optimizer=optimizer, loss='mean_squared_error', metrics=['mse', 'mae', custom_accuracy])
```

设置参数如下：

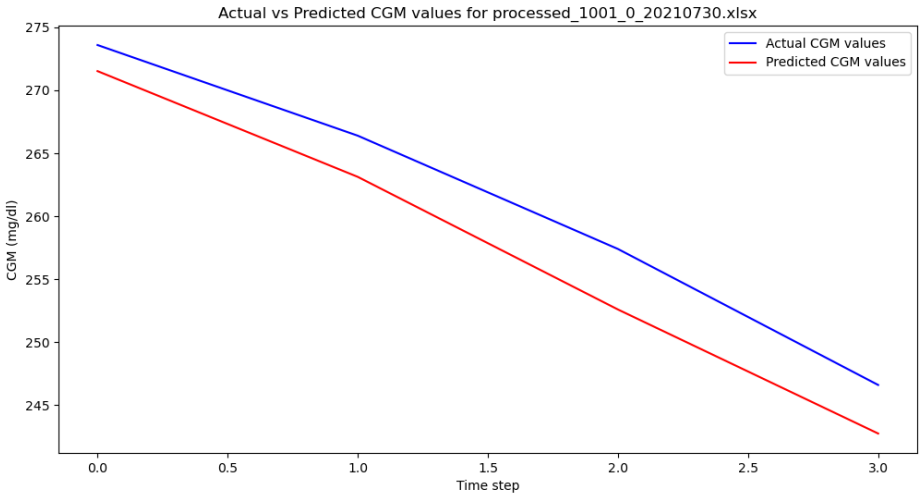
```
# 训练模型
model.fit(X_train, y_train, epochs=20, batch_size=32, validation_data=(X_val, y_val), callbacks=[metrics_history])
```

4.2.4 分析与可视化

仅对 T1DM 患者数据进行训练，并对于单个患者的数据集上进行测试，得到预测结果如下（由于篇幅原因仅展示部分病人结果）：



患者数据编号 1002_2_20210909 的 MAE: 0.9940147399902308
患者数据编号 1002_2_20210909 的 MSE: 1.3019271858886277



患者数据编号 1001_0_20210730 的 MAE: 3.5015296936035085
患者数据编号 1001_0_20210730 的 MSE: 13.242072255771994

MSE 和 MAE 的计算结果如下

数据集	MSE	MAE
训练集	0.0055	0.0503
验证集	84.7625	8.2482

由图表可知，模型在训练集上的预测和真实数据重合较多，说明模型已经较好地学习了训练集的性质。但如果观察其在验证集上的预测结果，可以发现模型预测与真实数据间的差异仍然较明显。训练集和验证集上的 MAE 差距巨大，所以，LSTM 模型可能存在数据过拟合，泛化能力不足的问题。而且，LSTM 在一些点的预测上和前一天的真实数据很接近，这种情况在 LSTM 的应用中十分常见。由于对血糖之类的时间序列数据，直接使用前一天的真实值预测后一天一般损失也较小。LSTM 能够较好地捕捉序列的趋势，但是却存在惰性，在真实应用时模型的有效性还有待进一步检验。

4.3 Transformer

4.3.1 原理

Transformer 模型是一个序列到序列的模型。与传统的循环神经网络（RNN）和长短期记忆网络（LSTM）不同，它是一种基于自注意力机制的架构，具有更好的并行运算能力以及可解释性。因此，Transformer 模型在一些特定时间序列预测场景应用中具有一定的适用性。

1) 多头注意力机制（Multi-Head Attention）

多头注意力机制是 Transformer 模型的核心组件。其基本思想是通过多个不同的注意力头，对输入序列进行多次线性变换和注意力计算，从而捕捉到不同子空间中的特征。这些注意力头的计算结果会被拼接起来，并通过一个线性变换层进行组合。

具体来说，多头注意力机制的计算步骤如下：

1. 线性变换：将输入序列通过线性变换得到查询（Query）、键（Key）和值（Value）向量。
2. 计算注意力得分：通过点积计算查询和键之间的相似度，得到注意力得分矩阵。
3. 应用 Softmax 函数：对注意力得分矩阵应用 Softmax 函数，将其转换为概率分布，得到注意力权重矩阵。
4. 加权求和：将注意力权重矩阵与值向量相乘，得到加权求和结果。
5. 拼接与线性变换：将所有注意力头的结果拼接起来，并通过一个线性变换层进行组合。

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1,\ldots,\text{head}_h)W^O$$
$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$
$$\text{Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

2) 位置编码（Positional Encoding）

由于 Transformer 模型没有内置的时间步信息，因此需要显式地加入位置信息。位置编码用于为序列中的每个时间步添加唯一的位置信息，使得模型能够捕捉序列中的时间依赖关系。位置编码可以是固定的，也可以是可学习的，常用的固定位置编码公式如下：

$$\mathrm{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$
$$\mathrm{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

3) 前馈神经网络（Feedforward Neural Network）

Transformer 模型中的每个编码器层包含一个多头注意力机制和一个前馈神经网络。前馈神经网络由两个线性变换层和一个 ReLU 激活函数组成，用于对注意力机制的输出进行进一步处理和特征提取。

4) 层归一化和残差连接（Layer Normalization and Residual Connection）

为了稳定训练过程并加速收敛，Transformer 模型在每个子层（多头注意力机制和前馈神经网络）之后都应用了层归一化（Layer Normalization）和残差连接（Residual Connection）。

5) Transformer 编码器层（Transformer Encoder Layer）

由一个多头注意力机制和一个前馈神经网络组成，每个子层后面都有层归一化和残差连接。多个编码器层堆叠在一起，形成 Transformer 编码器。

4.3.2 创建时间序列滑动窗口

滑动窗口技术是一种常用的数据处理方法，特别适用于时间序列数据。通过创建一个固定大小的窗口，滑动窗口可以从时间序列数据中提取出连续的子序列，这些子序列将作为模型的输入，用于预测目标变量。窗口大小表示每个训练样本的序列长度，即将要使用多少历史时间点的数据来预测未来的值。对每一个预测点，需要根据窗口大小取得它之前的历史数据作为特征，目标则是未来的 CGM 值。

```
# 创建滑动窗口
def create_windows(features, target, window_size):
    X, y = [], []
    for i in range(window_size, len(data)):
        X.append(data.iloc[i - window_size:i][features].values)
        y.append(data.iloc[i][target])
    return np.array(X), np.array(y)
```

在这段代码中，features 包含需要用来预测目标变量的特征列；target 是目标变量（即血糖水平）的列名；window_size 是滑动窗口的大小，即每个子序列包含的时间步数。

通过滑动窗口，可以将原始时间序列数据转化为一组样本，每个样本包含一个输入序列（长度为 window_size）和对应的目标变量。这样，时间序列的时间依赖性得以保留。

滑动窗口生成的输入序列和目标变量随后被用作模型的训练数据和验证数据。

```
# 创建数据窗口
X_dynamic_cat, _ = create_windows(dynamic_categorical_features, 'CGM (mg / dl)', window_size)
X_dynamic_num, y = create_windows(dynamic_numerical_features, 'CGM (mg / dl)', window_size)

# 重新整形数据
X_dynamic_cat = X_dynamic_cat.reshape(-1, window_size, len(dynamic_categorical_features))
X_dynamic_num = X_dynamic_num.reshape(-1, window_size, len(dynamic_numerical_features))

# 数据集划分
X_train_cat, X_val_cat, X_train_num, X_val_num, y_train, y_val = train_test_split(
    X_dynamic_cat, X_dynamic_num, y, test_size=0.2, shuffle=shuffle)
```

这段代码使用 create_windows 函数生成包含分类特征和数值特征的输入序列，以及对应的目标变量y;将生成的数据重塑为适合模型输入的格式;将数据集划分为训练集和验证集，用于模型训练和评估。

4.3.3 模型定义

在我们的血糖预测模型中，使用了时间序列的滑动窗口技术来构建输入序列。模型的输入包括分类特征和数值特征，通过嵌入层处理分类特征，通过线性变换层处理数值特征，然后将其输入到 Transformer 编码器中，结合时间变量进行特征提取和聚合。最终通过一个全连接层输出预测的血糖值。

我们在这里通过 Pytorch 来简单实现“Attention is All You Need”中描述的 Transformer 架构。因为是时间序列预测，所以注意力机制中不需要因果关系，也就是没有对注意块应用进行遮蔽。

1. 嵌入层（Embedding Layer）：将分类特征通过嵌入层转换为固定大小的向量表示。嵌入层是一种将离散变量映射到高维连续空间的技术，适用于处理具有多个类别的特征。
2. 线性变换（Linear Transformation）：将数值特征通过线性变换转换为与嵌入层相同维度的表示。这样可以确保分类特征和数值特征在同一空间中进行处理。
3. 编码器（Transformer Encoder）：将嵌入层和线性变换后的特征合并，并通过多个 Transformer 编码器层进行特征提取。Transformer 编码器层由多头自注意力机制和前馈神经网络组成，能够捕捉输入序列中各位置之间的依赖关系。
4. 输出层（Output Layer）：通过一个全连接层将特征映射到预测的目标值（即血糖水平）。

```python
class TimeSeriesTransformer(nn.Module):
    def __init__(self, num_categorical_features, num_numerical_features, num_targets, embedding_size, num_heads,
                 num_blocks, dropout_rate):
        super(TimeSeriesTransformer, self).__init__()
        self.categorical_embedding = nn.Embedding(2, embedding_size)  # Assuming binary categorical features
        self.numerical_embedding = nn.Linear(num_numerical_features, embedding_size * num_numerical_features)

        combined_features = embedding_size * (num_categorical_features + num_numerical_features)
        transformer_layer = nn.TransformerEncoderLayer(
            d_model=combined_features, nhead=num_heads, dropout=dropout_rate
        )
        self.transformer = nn.TransformerEncoder(transformer_layer, num_layers=num_blocks)
        self.out = nn.Linear(combined_features, num_targets)

    def forward(self, x_cat, x_num):
        x_cat = self.categorical_embedding(x_cat.long())
        x_cat = x_cat.view(x_cat.shape[0], x_cat.shape[1], -1)

        x_num = self.numerical_embedding(x_num)
        x_num = x_num.view(x_num.shape[0], x_num.shape[1], -1)

        x = torch.cat((x_cat, x_num), dim=2)
        x = x.permute(1, 0, 2)
        x = self.transformer(x)
        x = x.permute(1, 0, 2)
        x = x.mean(dim=1)
        return self.out(x)
```

Transformer 模型通过多头自注意力机制和前馈神经网络对输入序列进行建模。在训练过程中，滑动窗口生成的输入序列被输入到 Transformer 模型中，模型通过自注意力机制对序列中的每个时间步进行建模，捕捉时间步之间的依赖关系，从而实现对目标变量（即血糖水平）的预测。

4.3.4 模型训练与验证

在具体的训练中，我们以 80%和 20%的比例划分训练集和验证集。同时为了测试模型训练效果，我们使用平均绝对误差(MAE)来评估模型的性能。由于总共有 16+109=125 个 excel，我们就使用每一个 exccel 的最后四个数据点作为测试数据进行预测（15min，30min，45min，60min），计算平均 MAE 的值。

设置超参数如下：

```python
# 定义超参数
window_size = 12
embedding_size = 72
num_heads = 12
num_blocks = 8
dropout_rate = 0.1
batch_size = 16
learning_rate = 1e-5
shuffle = True
```

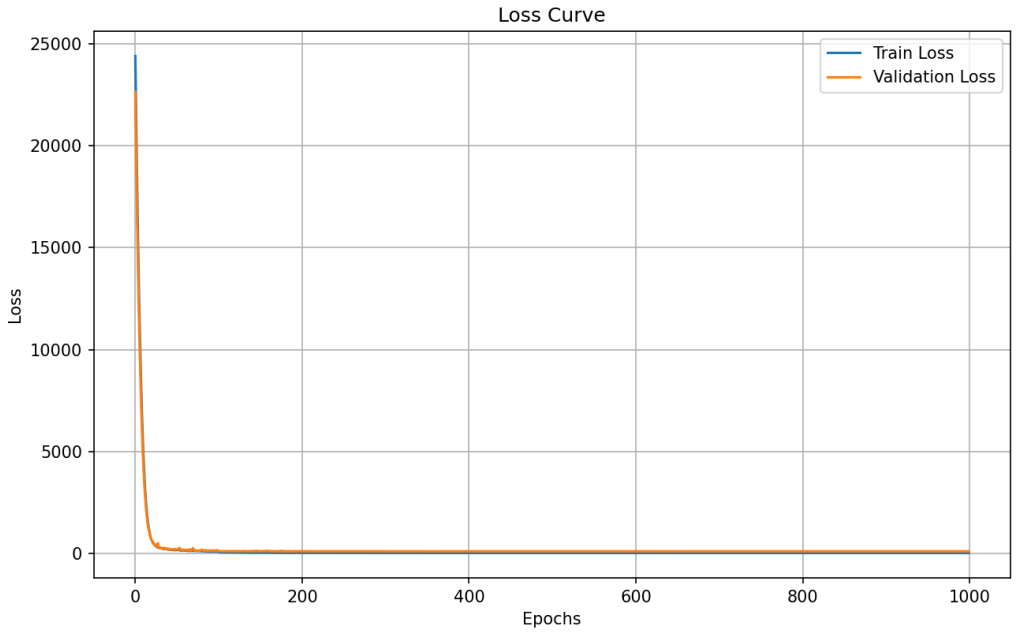
训练策略：

1) 损失函数：均方误差（MSE）损失函数，用于度量模型预测值与实际值之间的差异。
2) 优化器：AdamW 优化器，用于优化模型参数，学习率设置为 1e-5。
3) 学习率调度器：StepLR 调度器，每训练 10 个 epoch 后将学习率降低为原来的 0.1 倍。
4) 使用梯度裁剪（gradient clipping）来防止梯度爆炸。

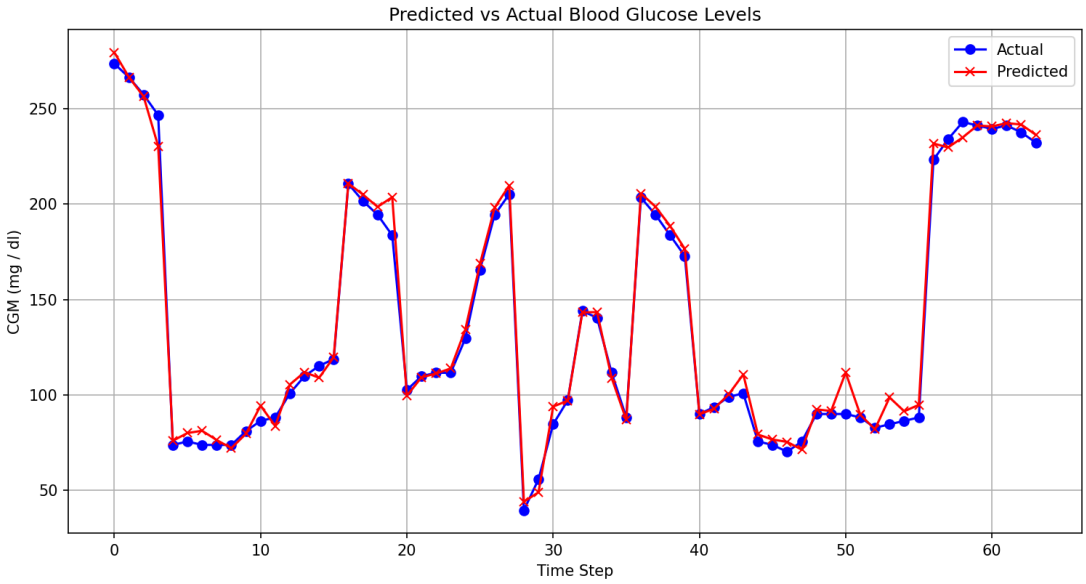
```
# 定义损失函数和优化器
criterion = nn.MSELoss()
optimizer = optim.AdamW(model.parameters(), lr=learning_rate)
scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=10, gamma=0.1)
torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=1.0)
```

4.3.5 分析与可视化

一、仅使用 T1DM 的数据进行训练的结果（epoch=1000）：



由图，训练集和验证集上的 loss 在 50 个 epoch 内即迅速下降至低位，达到了较优秀的效果。



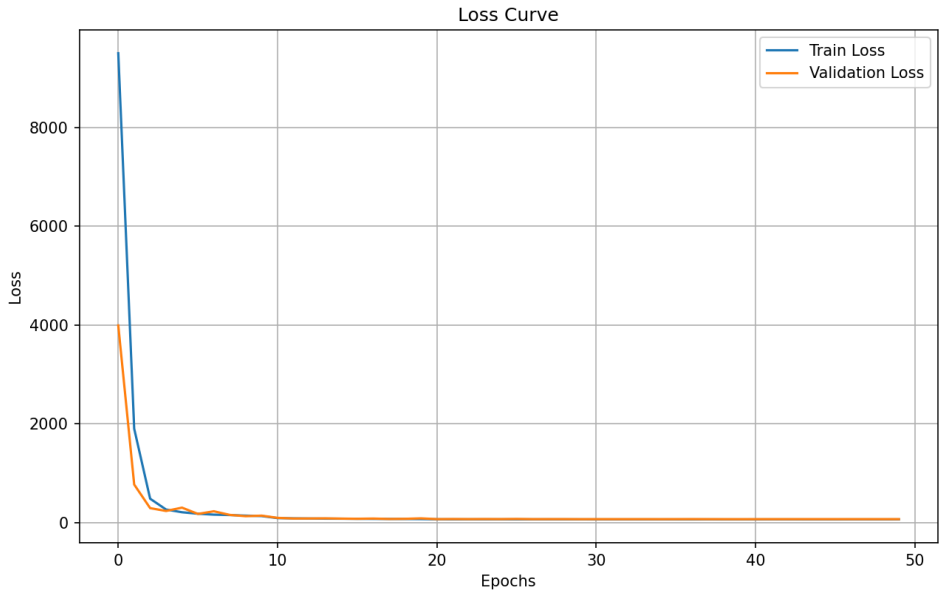
由图，使用 T1DM 中 16 个 excel 文件的最后四个数据点作为测试集，该模型比较能够反映出血糖数据的具体数值和大致变化趋势。

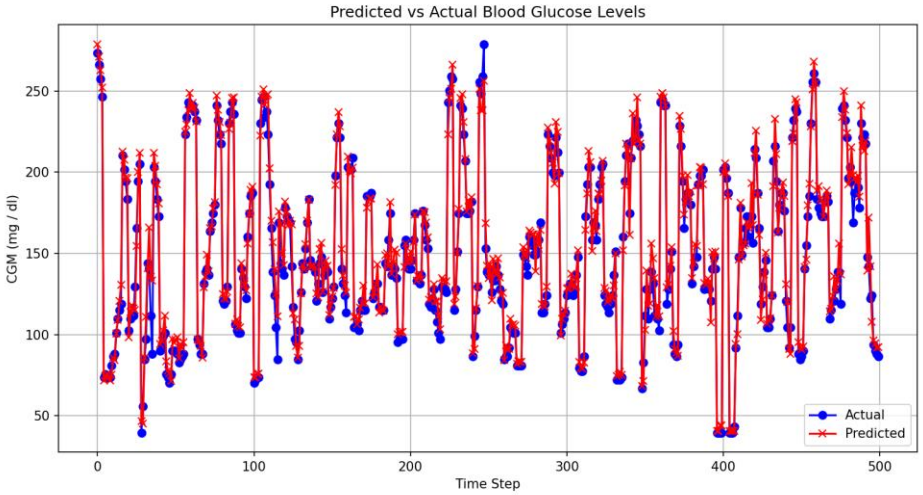
```
Predicted values:  [279.50516 266.47308 256.39426 230.34703]
Actual values:  [273.6 266.4 257.4 246.6]
Predicted values:  [75.98148 80.03243 81.20839 76.34853]
Actual values:  [73.8 75.6 73.8 73.8]
Predicted values:  [72.07781  79.654015 94.36191  83.730484]
Actual values:  [73.8 81.  86.4 88.2]
Predicted values:  [105.41841  111.939354 108.97053  119.79338 ]
Actual values:  [100.8 109.8 115.2 118.8]
Predicted values:  [210.80743 204.94678 198.55737 203.65817]
Actual values:  [210.6 201.6 194.4 183.6]
Predicted values:  [ 99.63209 109.17217 111.22865 113.87446]
Actual values:  [102.6 109.8 111.6 111.6]
Predicted values:  [134.4117  168.75832 198.06204 209.76831]
Actual values:  [129.6 165.6 194.4 205.2]
Predicted values:  [44.01653  48.975388 93.86938  97.00982 ]
Actual values:  [39.6 55.8 84.6 97.2]
Predicted values:  [143.22719 143.41423 108.87513  86.92886]
Actual values:  [144.  140.4 111.6  88.2]
Predicted values:  [205.59207 198.63248 188.47046 176.55779]
Actual values:  [203.4 194.4 183.6 172.8]
Predicted values:  [ 89.83263  92.73961 100.28112 110.7523 ]
Actual values:  [ 90.   93.6  99.  100.8]
Predicted values:  [79.25732  76.6151   75.37947  71.292366]
Actual values:  [75.6 73.8 70.2 75.6]
Predicted values:  [ 92.418434  91.58173  111.70064   89.585175]
Actual values:  [90.  90.  90.  88.2]
Predicted values:  [82.01051  98.66074  91.421455 94.78572 ]
Actual values:  [82.8 84.6 86.4 88.2]
Predicted values:  [231.75763 229.67712 234.94144 241.18346]
Actual values:  [223.2 234.  243.  241.2]
Predicted values:  [240.68678 242.43489 241.69418 236.2805 ]
Actual values:  [239.4 241.2 237.6 232.2]
```

打印所有预测结果和真实数据并进行计算。

在最后 4 个数据点上，该模型预测的平均 MAE 为 4.3196。

二、使用 T1DM 和 T2DM 的数据进行训练的结果（epoch=50）：





```
Predicted values:  [178.1013  188.22614 193.54648 185.18805]
Actual values:  [180.  187.2 187.2 176.4]
Predicted values:  [130.85077 120.16003  91.04354  87.96459]
Actual values:  [120.6 104.4  91.8 104.4]
Predicted values:  [219.00002 224.25569 244.85414 243.11606]
Actual values:  [221.4 232.2 239.4 237.6]
Predicted values:  [95.339226 92.31901  90.846535 92.53783 ]
Actual values:  [88.2 84.6 86.4 90. ]
Predicted values:  [141.57709 146.09323 179.84618 192.87492]
Actual values:  [140.4 154.8 172.8 185.4]
Predicted values:  [228.16583 251.13977 268.09024 254.78497]
Actual values:  [230.4 255.6 261.  255.6]
Predicted values:  [188.66876 191.5646  183.87343 182.56665]
Actual values:  [183.6 178.2 174.6 172.8]
Predicted values:  [172.53506 180.26274 188.89383 185.29831]
Actual values:  [172.8 181.8 185.4 181.8]
Predicted values:  [114.15971  115.628944 123.92993  135.76328 ]
Actual values:  [109.8 115.2 120.6 120.6]
Predicted values:  [127.48805 132.92561 156.00064 137.15643]
Actual values:  [127.8 138.6 138.6 118.8]
Predicted values:  [234.18825 249.78793 238.72891 224.1611 ]
Actual values:  [239.4 241.2 232.2 221.4]
Predicted values:  [192.51425 199.3706  215.39656 198.1174 ]
Actual values:  [196.2 203.4 194.4 169.2]
Predicted values:  [196.54868 193.5437  197.91707 184.82812]
Actual values:  [187.2 194.4 190.8 178.2]
Predicted values:  [241.31287 216.09323 212.94644 219.21928]
Actual values:  [230.4 221.4 223.2 217.8]
Predicted values:  [139.22665 172.14288 142.04807 107.98725]
Actual values:  [147.6 142.2 122.4 124.2]
Predicted values:  [96.5184   94.43569  91.71137  92.293495]
Actual values:  [93.6 90.  88.2 86.4]
```

由图，训练集和验证集上的 loss 在 5 个 epoch 内即迅速下降至低位，达到了较优秀的效果。使用 T1DM 和 T2DM 中的 125 个 excel 文件的最后四个数据点作为测试集，在最后 4 个数据点上，该模型预测的平均 MAE 为 5.9079。体现出该模型比较能够反映出血糖数据的具体数值和大致变化趋势。

```
Train Loss: 45.8540, Val Loss: 54.1766, MAE: 3.5159
Train Loss: 45.7730, Val Loss: 54.1766, MAE: 5.1967
Train Loss: 46.1209, Val Loss: 54.1766, MAE: 2.1710
Train Loss: 45.7747, Val Loss: 54.1766, MAE: 4.5597
Train Loss: 45.9453, Val Loss: 54.1766, MAE: 4.1278
```

打印出模型在训练集和验证集上的损失函数值，其差别相比 LSTM 较小，可得模型泛化能力较强。

五、对比总结，实验难点，改进建议

对比总结：

本研究通过比较支持向量机(SVM)、长短期记忆网络(LSTM)、时间序列 Transformer 三种不同的方法对血糖水平预测进行了系统的分析和对比。我们得到了以下主要结论：

1.支持向量机(SVM)：SVM 在处理小规模或中等规模数据集时表现良好，尤其是在高维空间中。然而，SVM 在本研究中的表现受限于特征选择和核函数的选择，需要进一步调整和优化参数以达到更佳的预测效果。

2.长短期记忆网络(LSTM)：LSTM 展示了其在处理时间序列数据方面的优势。在本研究中，LSTM 在训练集上表现良好，但在测试集上的泛化能力有待提高。这表明过拟合是 LSTM 需要进一步解决的问题。

3.时间序列 Transformer：时间序列 Transformer 通过引入深层结构，能够学习到数据中的复杂模式和关系。时间序列 Transformer 不仅在训练集上取得了良好的结果，也在测试集上也显示了更强的泛化能力，这表明其在处理具有时间相关性的血糖数据上的潜力。

综上所述，虽然每种模型都有其优点和局限，但时间序列 Transformer 在本研究中表现最为出色。未来的研究可以探索将时间序列 Transformer 与其他技术（如波动性建模）结合的混合模型，以进一步提高血糖预测的准确性和泛化能力。此外，增加数据量和引入更多类型的数据也可能有助于提高模型的预测性能。

实验难点和缺陷：

在数据预处理和特征工程方面，由于技术能力的限制，我们的方法是较为简略的。我们的原意是根据食物品种和重量来计算具体的碳水化合物摄入，根据服药品种和剂量来量化服药对患者的影响，但由于复杂门类的食物和药物，以及数据质量的问题，我们的设想难以推进。

对于动态特征，我们使用上个时间段和下个时间段的平均值来填充空缺数据行；而对于静态特征，我们直接删去了存在空缺数据的特征列，而此时 T1DM 比 T2DM 多五列完整的静态特征数据，所以我们只能先对 T1DM 进行训练尝试，然后再把它们合并，将其特征数量调整至相同，对合并后数据进行训练。在这里，我们非常有可能遗漏了对于血糖水平非常重要的静态特征。

其次，在模型的选择上，我们没有尝试模型的多元融合，没有去魔改并添加一些新兴的模型架构。同时，由于受制于个人电脑的 GPU 性能导致训练速度缓慢，我们难以设置更多的层数和特征数，由于内存较小，我们也无法加载太大 batch_size。

在时间序列Transformer 模型中，我们已经通过使用不同的处理方式(嵌入层和线性层)来区分处理分类特征和数值特征，但可以进一步尝试探索几种创新方法。可以使用更复杂的特征组合方法来捕捉分类特征之间的交互作用，例如特征交叉；在处理分类特征时，可以设计专门的注意力机制来加强模型对这些特征的学习，类似于多头注意力机制中不同头关注不同的信息；根据类别特征的重要性和多样性动态调整嵌入维度。在 LSTM 模型中，可以通过加入注意力机制，可以帮助模型关注输入序列中与预测目标更相关的部分。同时也可以构建多个 LSTM 模型，使用不同的模型结构、超参数或特征组合训练多个 LSTM 模型。

改进建议：

特征的选择和特征数量对于模型的预测效果有很大的影响，为此我们小组使用了两种不同的方法来测试特征对模型的重要程度，需要说明的是，为了减少模型的计算时间，我们仅使用最原始的 LSTM 模型进行测试。

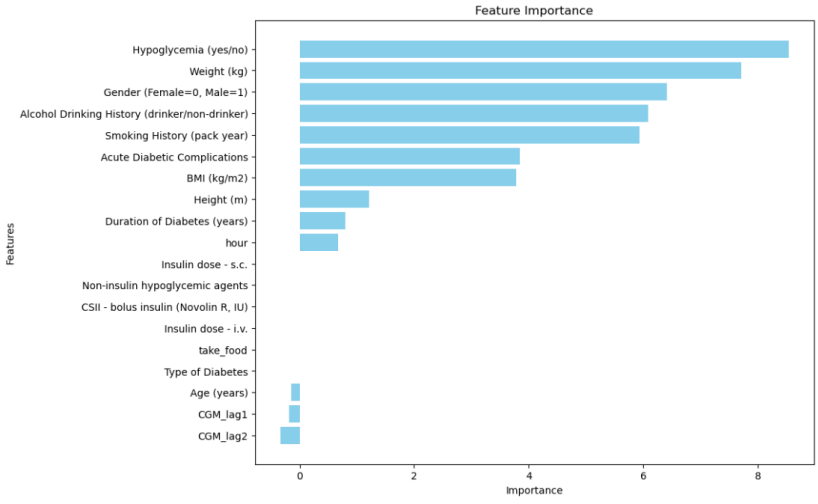
第一种方法是基于特征置换重要性的方法，其基本思想是通过置换特定特征的数据，观察这种置换对模型性能的影响来衡量该特征的重要性。如果置换某个特征的数据导致模型性

能显著下降，那么这个特征被认为是重要的；反之，如果置换某个特征的数据对模型性能影响不大，那么这个特征的重要性较低。

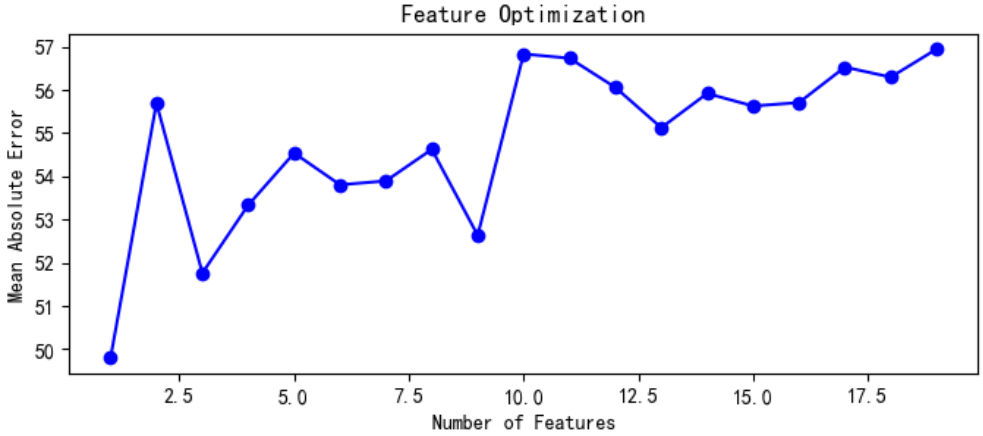
我们首先使用完整的测试数据集，计算模型的基线性能指标（MAE、MSE），然后对于每个特征，打乱该特征的数据（即随机置换该特征的取值），并保持其他特征值不变，接着使用置换后的数据集，进行模型的性能指标的计算，最后计算特征的重要性。其中特征重要性定义为置换后的性能与基线性能之间的差值。如果置换特征后的性能显著下降，说明该特征重要性较高;如果置换特征后的性能变化不大,说明该特征重要性较低。在进行实验之后，结果如下：

```
Feature: Hypoglycemia (yes/no), Importance: 8.536023330310044
Feature: Weight (kg), Importance: 7.710996857522034
Feature: Gender (Female=0, Male=1), Importance: 6.4083203989361905
Feature: Alcohol Drinking History (drinker/non-drinker), Importance: 6.092310790410117
Feature: Smoking History (pack year), Importance: 5.930366697765528
Feature: Acute Diabetic Complications, Importance: 3.8496448373037637
Feature: BMI (kg/m2), Importance: 3.782238063358122
Feature: Height (m), Importance: 1.2127028064122243
Feature: Duration of Diabetes (years), Importance: 0.8000169171227185
Feature: hour, Importance: 0.6755406470525855
Feature: Insulin dose - s.c., Importance: 0.0
Feature: Non-insulin hypoglycemic agents, Importance: 0.0
Feature: CSII - bolus insulin (Novolin R, IU), Importance: 0.0
Feature: Insulin dose - i.v., Importance: 0.0
Feature: take_food, Importance: 0.0
Feature: Type of Diabetes, Importance: 0.0
Feature: Age (years), Importance: -0.1408087881784681
Feature: CGM_lag1, Importance: -0.18539817749507392
Feature: CGM_lag2, Importance: -0.33464960597810034
```

将得到的特征重要性分数通过柱状图表示出来，如图所示：



可以看到，特征 Hypoglycemia、Weight、Gender、Alcohol Drinking History 以及 Smoking History 对模型的影响程度比较大；而 take_food、Insulin dose、CSII - bolus insulin 等特征对模型的影响程度几乎为 0，这与我们现实生活中的认知相悖，我们猜测可能是因为 take_food、Insulin dose、CSII - bolus insulin 这些特征间相关度比较高，因此对模型的预测没有贡献，可以考虑减少这其中的一些特征。为对特征进行优化，我们把这些特征按照重要性的高低进行了排序，然后依次将这些特征加入到模型中，从而得到模型预测水平与特征数量的关系，如图所示：

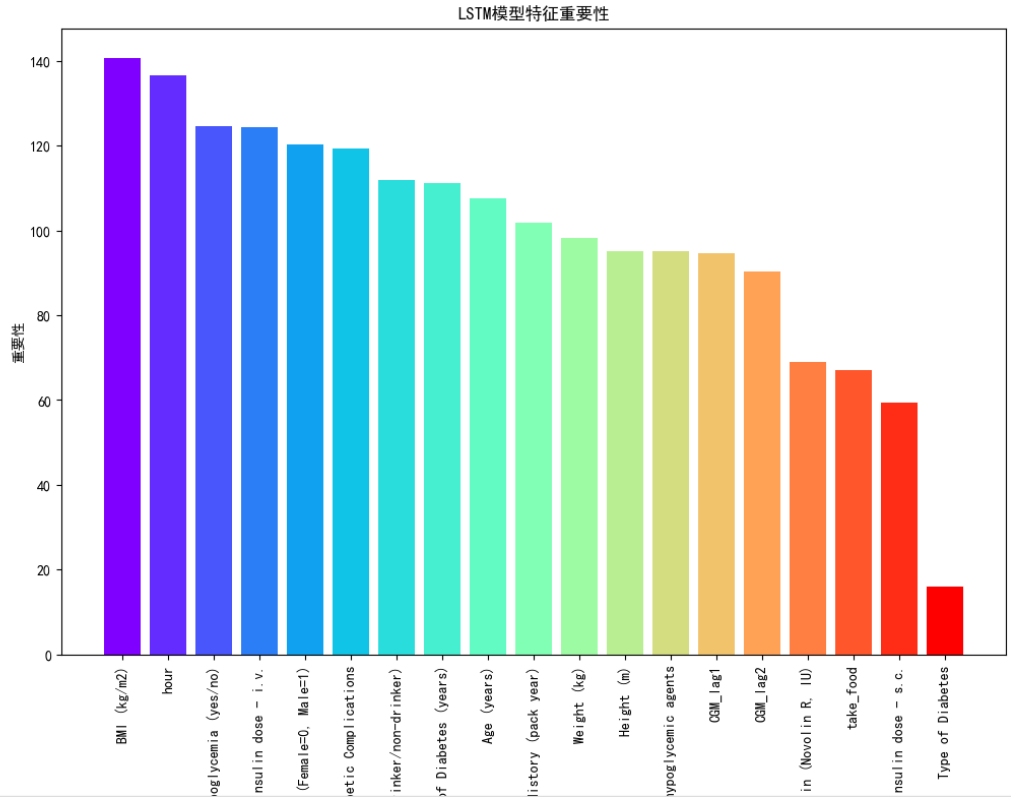


可以看到，不考虑极端的特征数量较少的情况，模型在特征数量为 9 的时候 mae 的值最低，说明我们可以考虑在已经选择的特征中去掉一部分预测效果重复的特征。

第二种方式是提取 LSTM 层的输入权重，并通过计算权重的绝对值总和来估计特征的重要性，LSTM 层学习到的权重一定程度上反映了不同特征在不同时间步上对预测的贡献程度。于 LSTM 是一种特殊的递归神经网络，用于处理序列数据，其通过输入门、遗忘门和输出门来控制信息的流动，以捕捉长时间依赖关系；此外在 LSTM 层中，每个输入特征都有一个对应的权重矩阵，这些权重决定了特征对 LSTM 单元输出的影响，因此可以通过聚合这些权重的绝对值来量化每个特征的重要性。在进行实验之后，得到的结果如下：

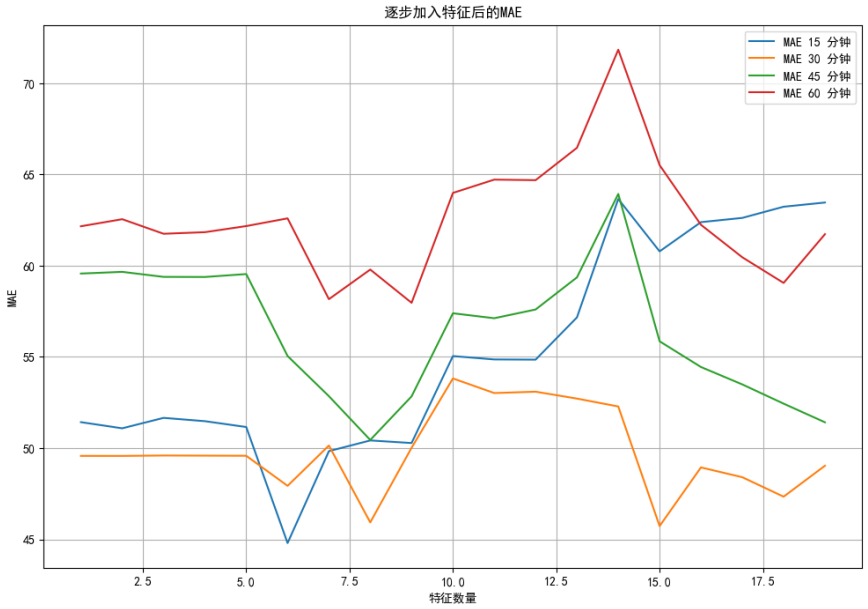
```
特征重要性：
BMI (kg/m2): 140.6364
hour: 136.6935
Hypoglycemia (yes/no): 124.6119
Insulin dose - i.v.: 124.3774
Gender (Female=0, Male=1): 120.2913
Acute Diabetic Complications: 119.3769
Alcohol Drinking History (drinker/non-drinker): 111.8861
Duration of Diabetes (years): 111.2436
Age (years): 107.5265
Smoking History (pack year): 101.9035
Weight (kg): 98.2891
Height (m): 95.0352
Non-insulin hypoglycemic agents: 95.0069
CGM_lag1: 94.6409
CGM_lag2: 90.2883
CSII - bolus insulin (Novolin R, IU): 69.0061
take_food: 66.9758
Insulin dose - s.c.: 59.5000
Type of Diabetes: 15.9928
```

将得到的特征重要性进行可视化，如下图所示：



可以看到，BMI、hour、Hypoglycemia 等特征的重要性比较高，而 Type of Diabetes 的排名是最低的，这也与实际上 T1DM 糖尿病患者的情况比较符合。T1DM 患者由于体内缺少产生胰岛素的细胞，因此需要通过注射胰岛素来降低血糖的水平，而无论患者摄入的食物是成分如何，体内系统都无法针对血糖的水平进行血糖调节，因此这个结果也比较符合科学常实。

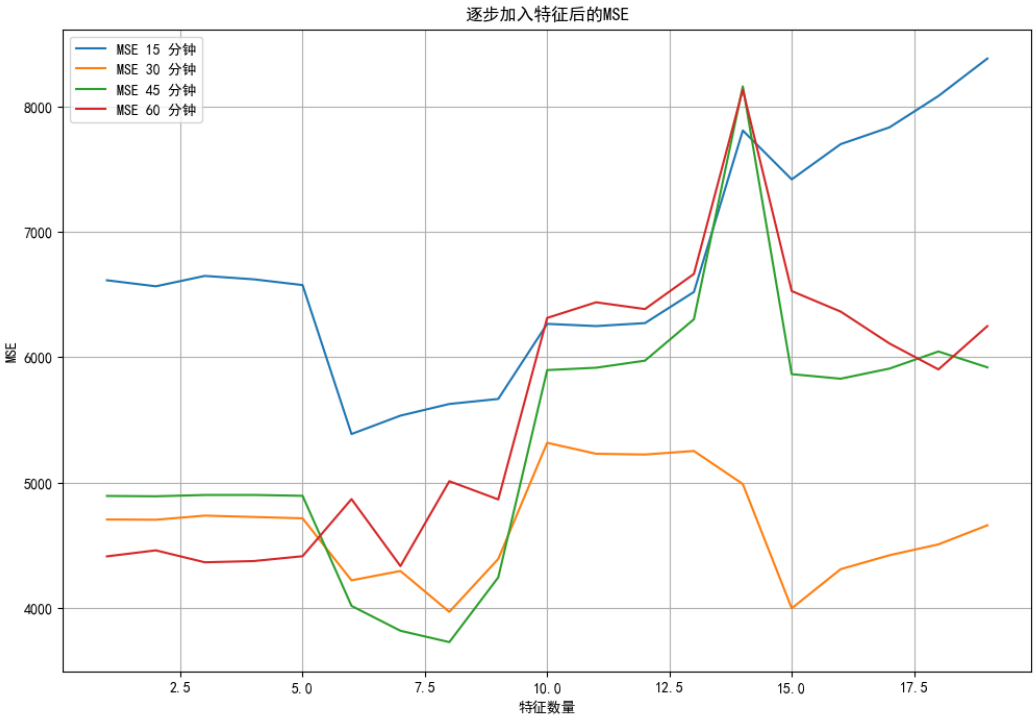
为了更加具体地分析不同时刻下血糖水平的变化与特征数量的关系，我们在第二种方法中比较了不同特征数量在不同时间段下的整体预测水平，如下图所示：



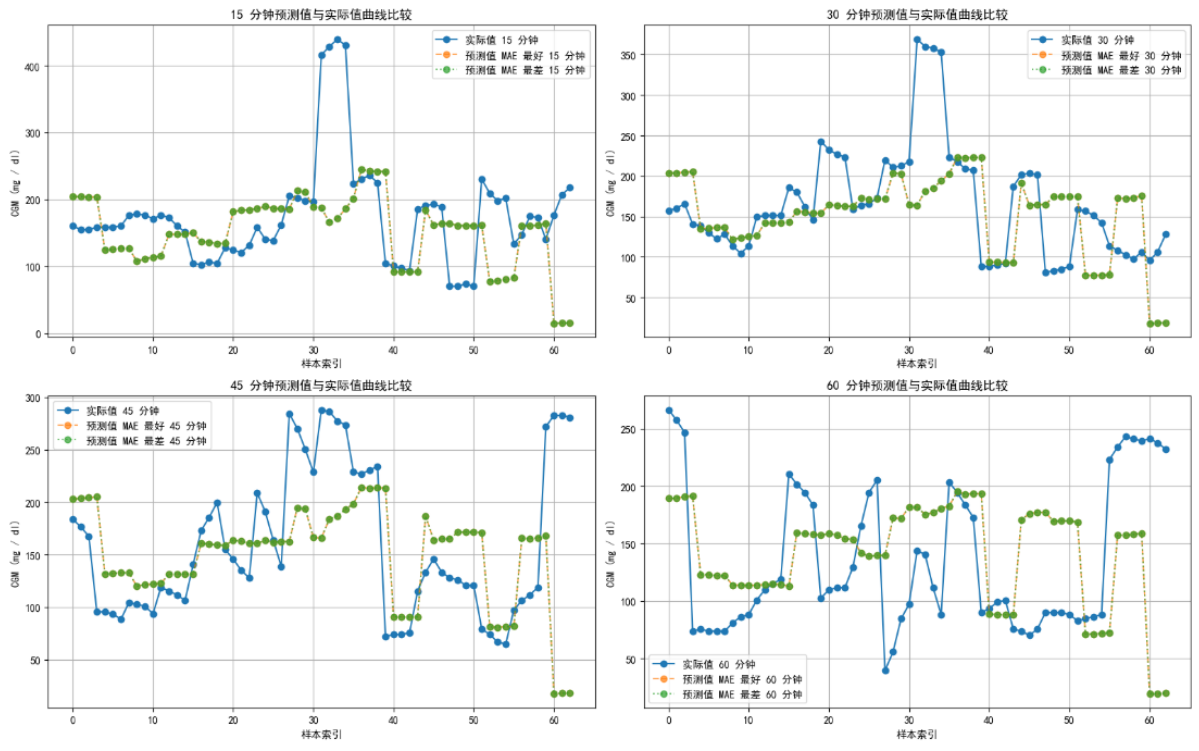
可以看到四条 MAE 曲线的整体走势是一致的，除了 30 分钟时间段在特征数量为 15 时出现了显著的下降，但是不同时间段最低 MAE 是在不同数量特征下取得的，说明某些特

征变量对血糖水平的扰动作用，比如说在午餐、晚餐的时间段往后的一个小时患者的血糖水平会受到 take_food 特征的剧烈影响，在先前观察患者数据时可以注意到患者的进食时间一般集中于中午的 11 点-11 点 30 分之间，因此进食这个特征可能就会影响到 15 和 30 两条曲线，在加入 take_food 这个特征后这两条曲线可能会出现相同的上升趋势（图中刻度 17 往后部分）

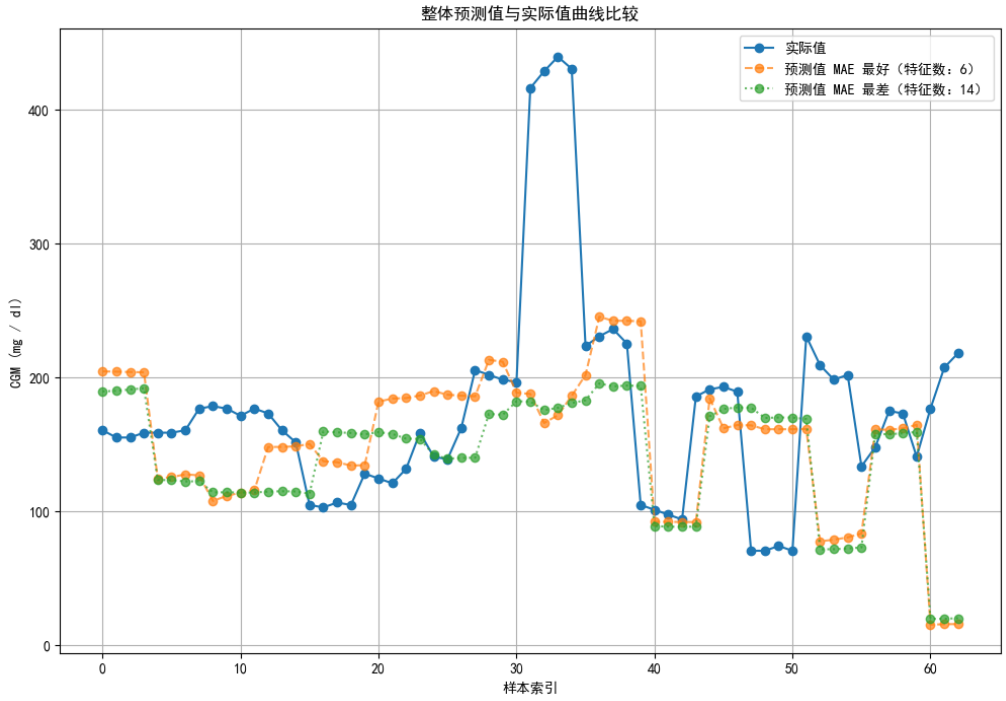
此外，我们小组还对 MSE 的值也进行了可视化，可以从图中得到相似的结论：



为比较预测最好的 MAE、实际水平、最差 MAE 之间的关系，我们按照时间段对这三个指标进行了可视化：



可以看到不同时间段下最好和最差的 MAE 两条曲线几乎重合，而这两条曲线与实际的水平有明显的差距，出现这样的原因一个可能是因为我们的特征还不够完美，需要作进一步的优化。比如说将一些特征进行交互，把既吸烟又喝酒的特征构造出来，同时对于一天不同的时间段，人的血糖水平会受到一些其它因素的影响，可以考虑将进食的特征和注射胰岛素的特征进行交互；同时可以将 BMI、时间 hour 这些特征进行分类，分类出高、中、低 BMI 患者和早、中、晚这些时间段。最后我们小组为也考察了最优和最差 MAE 在 T1DM 患者中的整体水平：



可以看到不同个体间的血糖水平还是有比较明显的差异，出现了三条比较明显的曲线。

附录：

我们在数据预处理的过程中，遭遇了以下的一些 bug，可供老师作为参考，进一步增加 Chinese diabetes datasets for data-driven machine learning 的适用性和健壮程度。

- T2DM 中有两个表格的特征列名都没有单位。
- T2DM 的第二个表格 CSII - bolus insulin (Novolin R, IU)没有逗号
- T2DM 的 2029_0_20210526.xlsx 的 2021/6/3　9:29:00 时刻血糖值缺失
- T1DM 的 Duration of Diabetes (years)多了一个空格
- T2DM 的 Duration of diabetes (years)的 d 没有大写