数据仓库与数据挖掘大作业

软博 22 管浩荃 2022312726, 软硕 221 陈子陵 2022213872

1 数据处理

可以发现，为了基于脓毒症患者在 24 小时观测窗口的生命体征数据预测观测窗口结束之后 6 小时是否死亡，首先需要保证患者在观测窗口内的数据是干净且有效的。因此，需要对时间序列做出相应的处理。

1.1 数据缺失值处理

对于数据的缺失值，在处理的过程中可以发现有空缺的行较少，因此将此类数据全部去除。事实上，对于训练数据集而言，去除缺失值前后的数据集维度分别为 $(265030, 7)$ 和 $(214265, 7)$。被去除的行数相比于总行数的比例并不高。

1.2 异常值处理

通过对数据集内数据的观察，可以发现数据集中有一些不符合实际的极端数值，即病人的生命体征不可能处于该值下。因此，我们将其识别为异常值。

结合实际情况和文档给出的健康范围，我们去除了如下数据所在的数据行：呼吸率超过 60；氧饱和度低于 50；平均动脉压 < 0 或 > 180。

同时注意到，呼吸率为 0 有可能有两种情况。其一，在数据收集的过程中，由于机器故障或传输问题，呼吸率并没有被成功接收。其二，可能病人在该时间点已停止呼吸，呼吸率为 0 是病人的真实情况。基于如上考虑，在数据处理的过程中只对呼吸率过高的数据所在行进行处理。

1.3 不等长序列处理

可以发现，对于每个病人，其时间序列虽然都处于 24 小时的观测窗口之内，但是序列的长度并不相同。为保证能使用 batch 进行模型的训练，并且规范所有病人的生命体征数据序列长短以优化模型效果，拟对所有病人的序列进行如下处理。

1. 假如原本序列长度超过 20，那么只保留最后的 20 个时刻；
2. 假如原本序列长度不足 20，那么将最后一个时刻的数据进行重复。

1.4 规范化

由于患者的生命体征数据所处的范围并不相同，因此不进行规范化可能会使得每个字段对模型的影响有极大的差别。因此，需要将所有的数据进行规范化，以平衡四个字段的影

响。由此，对于四个字段 heartrate,resprate,map,o2sat 中的每一个，都进行如下处理 (下面用 attr 表示前述任意字段)：

1 首先，在数据集中加入了新字段 attr_DELTA 表示差分后的结果。这是出于两个原因：(1)、体征稳定的病人更加安全；(2)、病人本身有自身体质，比如说自然状态下的某些病人心率就偏低，所以用差分后的数据帮助学习。

2 而后，在数据集中加入了新字段 attr_ERR 表示异常程度：为了利用说明文档给出的正常值范围，新增该字段表示属性偏离正常范围的程度。

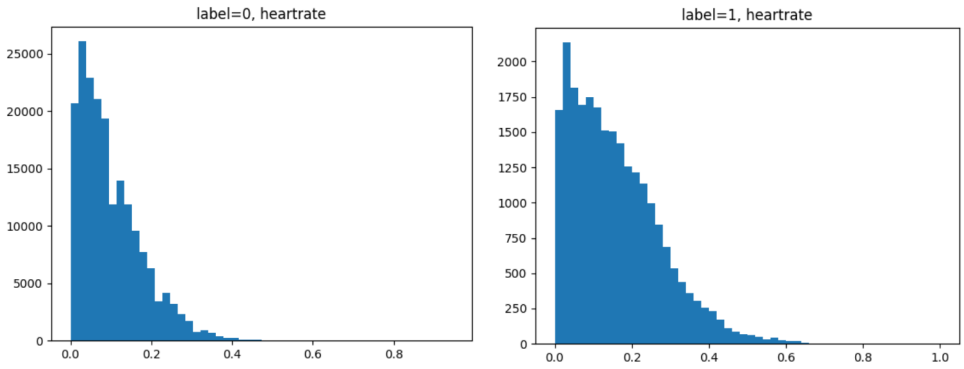
2.1 注意到许多时候病人指标会在正常范围边缘 (比如氧饱和度在 94 ~ 96 之间波动而正常边界是 95)，所以不是简单的 0/1 表示是否正常而是用连续值，指标偏离正常范围越多那么该字段数值就越大。

2.2 根据背景知识，氧饱和度指标是越高越好 (100 是最好的)。

3 将 attr 从原始数据改为数据离正常范围中间值的距离。这是因为原始数据太小和太大 (比如心率过低和过高) 都是不利生存的表现，做这一步处理之后，数值就是越小越好，方便学习。

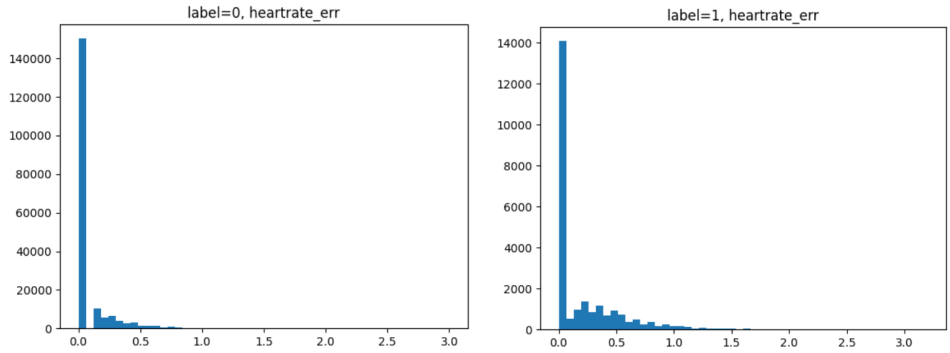
4 对数值型数据的规范化流程，使用正常值范围对数据进行标准化，将正常值的范围变为 $[0, 1]$。这样可以确保模型正常学习。

处理后的部分字段分布图所示。其中，左右两图分别表示标签为 0 和 1 的训练集数据。在对数据集进行预处理后，可以发现数据大小和生存情况的关系更加直观。



2 回归模型与实验结果

本节中，我们选择了传统 RNN，GRU，LSTM 进行回归预测，并用 Accuracy、F1 分析预测结果，并给出了 ROC 曲线和混淆矩阵。这是因为 Accuracy 可以反映出模型在整个数据集上的表现，F1 可以反映出模型在正样本和负样本上的综合表现，ROC 曲线和混淆矩阵可以体现出模型预测标签和实际标签之间的关系。



2.1 传统 RNN

我们使用 torch.nn.RNN 作为模型对数据集进行训练，如图 1所示。可以发现，对于传统 RNN，随着 epoch 轮数的增进，Accuracy 和 F1 Score 变化不明显。这说明 RNN 对数据集的建模能力不强。Accuracy 虽然达到了 0.91，但是其仍低于 GRU 和 LSTM 的结果，且这是因为数据集中有很多标签为 0 的数据。同时，F1 Score 只有 0.4 左右，这是较低的。

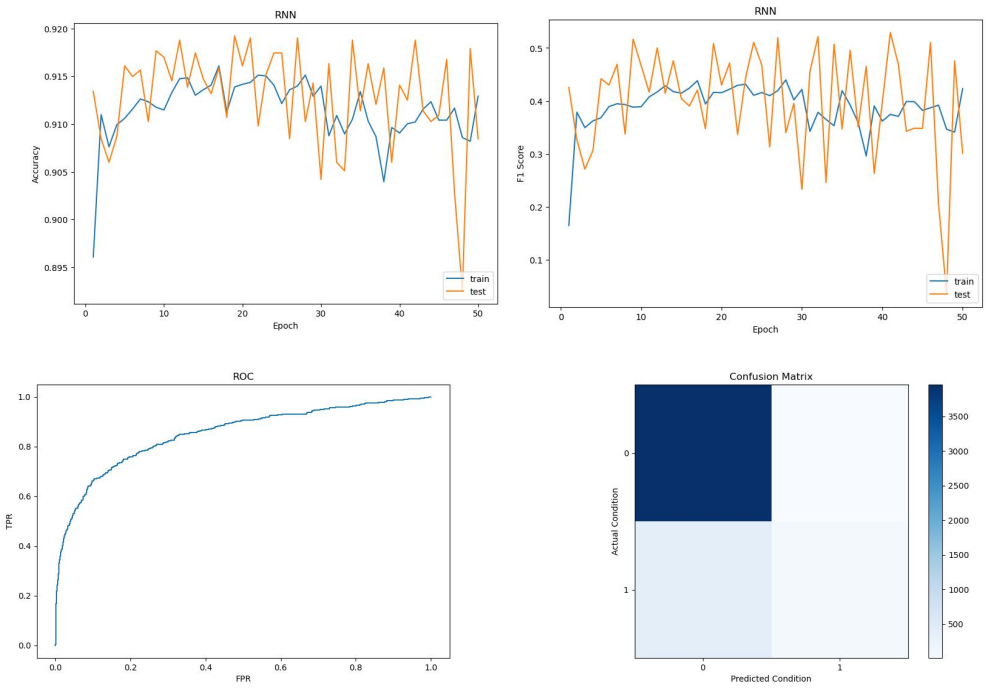


图 1: RNN

2.2 GRU

2.2.1 机器学习库函数

我们使用 torch.nn.GRU 作为模型对数据集进行训练，如图 2所示。可以发现，GRU 对训练集数据的建模能力强，可以在 50 个 epoch 处达到极高的 Accuracy 和 F1 Score。但是

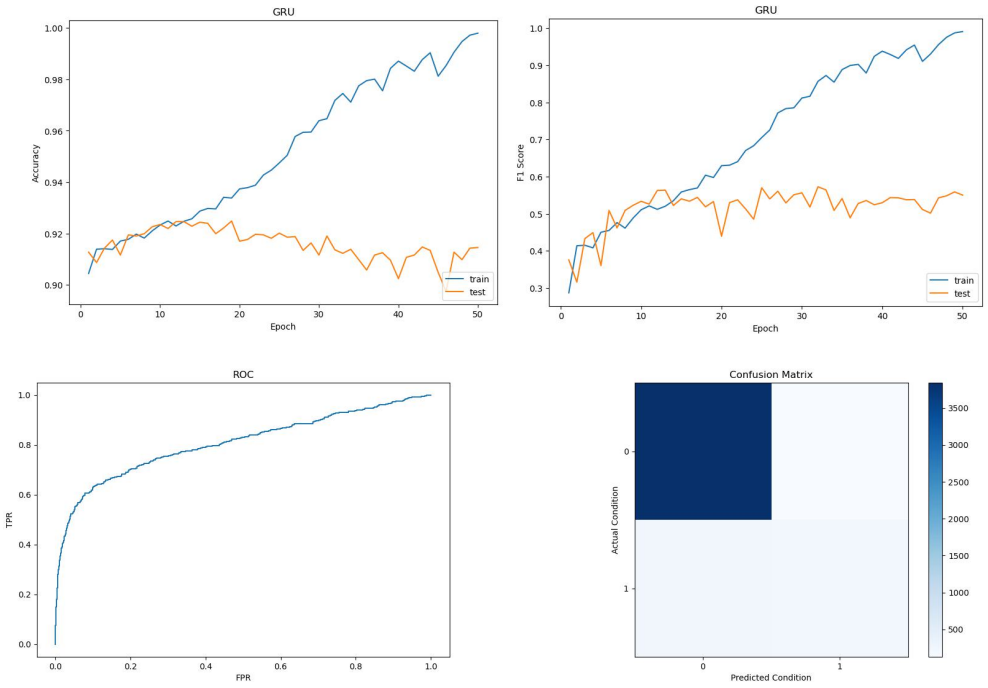


图 2: GRU（torch.nn）

可以发现，虽然测试集的 F1 Score 随着 epoch 而上升，但是其最高值仍然在 0.58 左右。这说明模型存在过拟合，且 recall 较低。这从 ROC 曲线课混淆矩阵中也能看出。

2.2.2 人工实现

对于人工实现的 GRU，我们使用 torch 中提供的 tensor 乘法等方法进行模型的构建，如图 3所示。可以发现，Accuracy 和 F1 Score 的变化趋势与调用机器学习库函数的结果相同，但是训练速度慢于调用库函数的场景。这是因为库函数中使用了多种加速和优化方法，而我们的人工实现则并没有使用。同时，可以发现事实上测试集的 F1 Score 与调用库函数的结果相似。这可能是因为更少的优化方法反而带来了更少的过拟合。

2.3 LSTM

我们使用 torch.nn.LSTM 作为模型对数据集进行训练，如图 4所示。对于 LSTM，可以发现其训练速度也略慢于 GRU。同时，其 F1 Score 与 GRU 相近，且优于传统 RNN。这说明，LSTM 在未知长度的时间序列建模任务上仍然有优势，且过拟合的现象相较于 GRU 并没有那么严重。但是 LSTM 在 Recall 上的表现仍然不佳，这可以从混淆矩阵中看出。(图在最后一页)

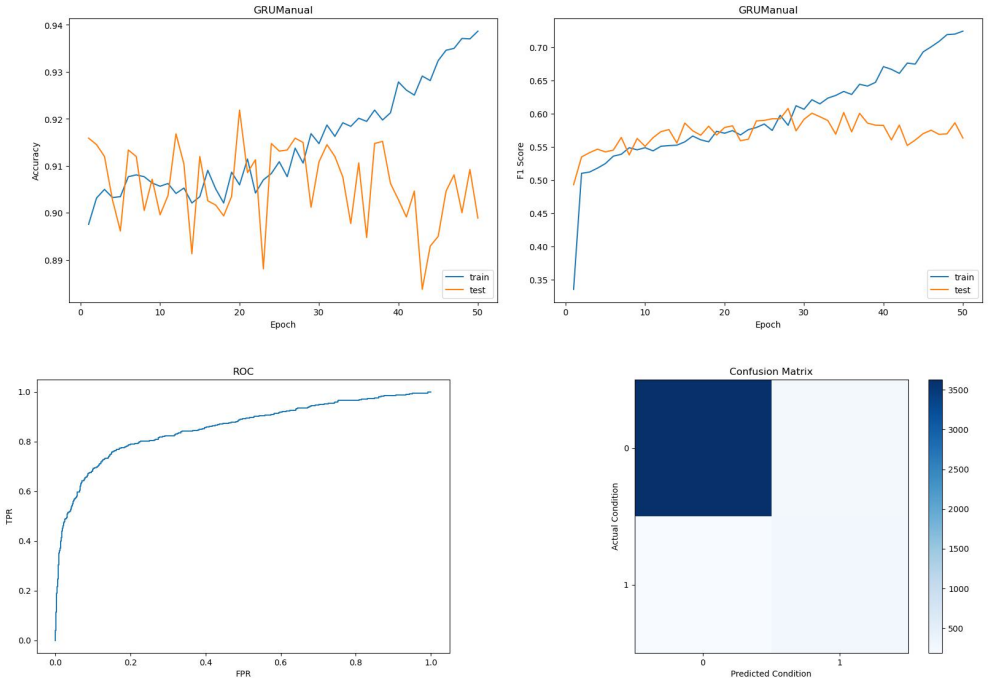


图 3: GRU（人工实现）

3 超参数修改

修改学习率为 0.05（原本为 0.002），可以得到四个模型的结果如图 5所示。由于篇幅限制，只放置 F1 曲线的变化。可以发现，LSTM 和传统 RNN 均对模型建模失败，表现出来 0 的 F1 Score。这是因为模型将所有的标签全部预测为了负值，从而使得 Precision 和 Recall 都为 0。对于 GRU，其训练效果仍然远逊于 0.002 的学习率。因此，可以发现不能将学习率设置得过大。

4 分工说明

所有部分均由二人协作完成。

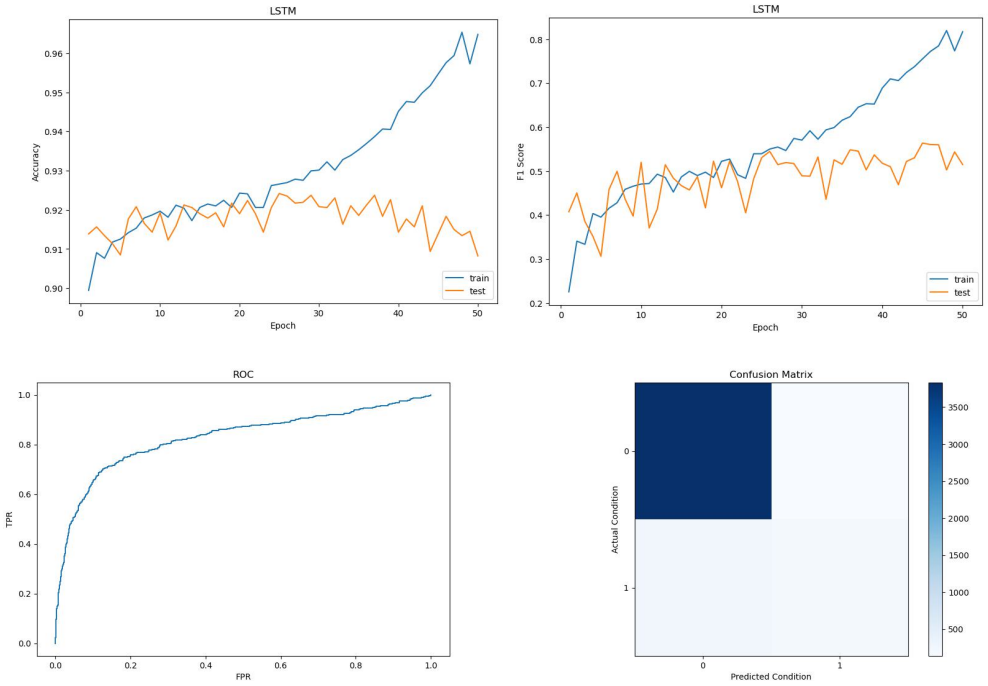


图 4: LSTM

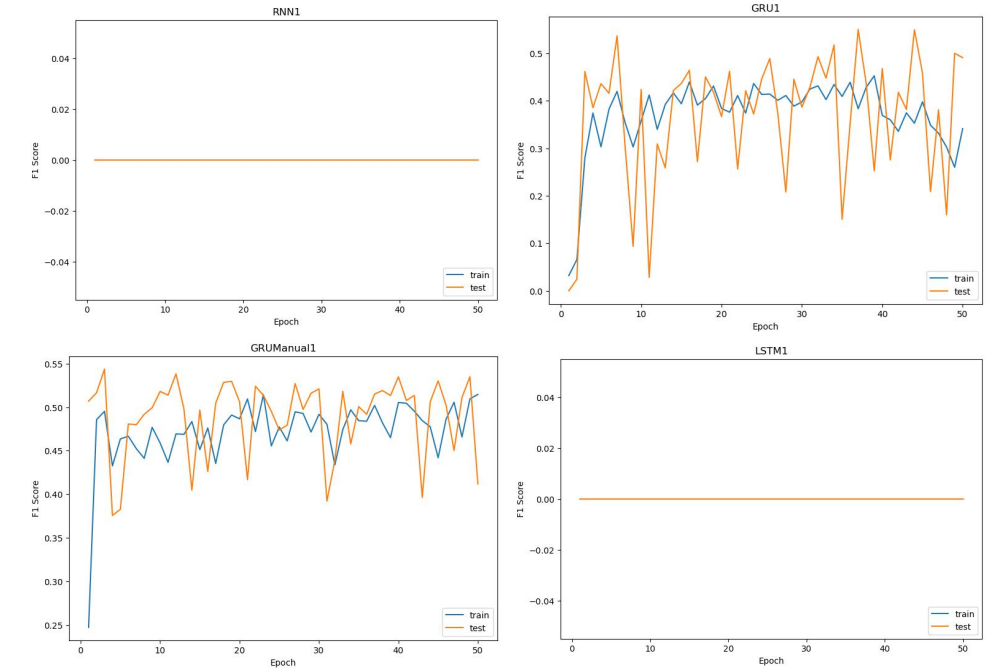


图 5: 学习率 0.05