**异常检测项目报告**

我们的地下工程健康检测系统需要对于地下工程检测产生的一些指标，诸如温度，湿度，气压，进行合理的预测和异常检测，并且最终返回到我们的项目界面中，便于后续的工程维护。

异常检测的工作，我们时序异常检测框架，他是一个多模型的集成异常检测模块，主要分为：构建时间序列、选择时间序列模型进行数据训练、对训练后的时间序列进行异常检测，三个模块。

基于此框架的异常检测功能，相较于传统的工程中使用的阈值检测模型，拥有多方面的优点。是一种更符合现代化数学统计预测思想的，结合计算机强大计算能力的综合异常检测模块，是我们工程中的创新点、亮点之一。

其中，很关键的一部分就是时间序列预测模型。时序异常检测框架中提供了很多种预测模型，集中放在tsmm模块中并且可以在配置文件中进行选择。下面是我们项目中运用到的时间序列预测模型介绍。

**1 时间预测序列模型框架介绍：**

我们采用的时间序列预测模型框架主要分为两个模块：tsmm时间序列建模模块和adm异常检测模块。

系统将输入数据转化为时间序列，然后选择一个时间序列模型应用于时间序列，构成时间序列建模模块（TSMM），再把异常检测模型应用于时间序列建模模块输出的预测值，构成异常检测模块（ADM），最终把检测出的异常点写入需要返回到我们工程前端界面的数据结构里。

一开始的测试方式中，我们采用了将原始数据写入CSV文件的方式来对文件中数据进行操作，后来我们有了自己的mqtt报文数据之后，改为访问数据接口API的方式来对数据进行获取和访问，关于数据来源的操作，可以在配置文件中进行修改。

具体到模块工程中，步骤可以分为以下四个步骤：

1.1 初始化配置文件和输入文件

主函数加载配置文件 sample\_config.ini 和服务器端数据获取接口http://192.168.100.100:8080/messages?measurement=/SenseHatB/socc3dzf83ty/Temperature&period=7200，来获取两个小时内的温度或各种类型的传感器数据，并实例化输入文件的处理对象APIInputProcessor。

1.2 把输入文件转化为TimeSeries列表

APIInputProcessor负责把输入文件转化为List<TimeSeries>，其中一个TimeSeries对象存储了一列时间戳及对应的一列值，若输入文件若包含了一列时间戳和n 列值，那么就会产生n个TimeSeries组成的列表。

同时TimeSeries也封装了时序数据的元数据信息，如文件名，列名称等。然后根据配置中的参数，判断是否需要对输入数据进行简单的缺失值处理和聚合处理。

1.3 加载时序模型和异常检测模型

使用ProcessableObjectFactory工厂类构建时间序列模型。时序模型通过类加载器加载并实例化，实际使用的实例化的类为AutoForecastModel, 然后把时间序列TimeSeries和时序模型List<TimeSeriesModel>封装成ModelAdapter，作为时序模型的模块。

使用ProcessableObjectFactory工厂类构建异常检测模型。异常检测模型通过类加载器加载并实例化，示例中实例化的类为ExtremeLowDensityModel, 然后把时间序列TimeSeries和异常检测模型List<AnomalyDetectionModel>封装成AnomalyDetector，作为异常检测模型的模块。

最后把时序模型的模块ModelAdapter和异常检测模型的模块AnomalyDetector封装成异常检测处理模块DetectAnomalyProcessable.

1.4 使用时序模型预测，并使用异常检测模型检测出异常点

先使用时间序列训练模型，根据训练好的模型得到预期值，这里使用的模型为AutoForecastModel。接着使用异常检测模型计算若干项误差指标对应的阈值，其中误差指标包括：MAE, sMAPE, MAPE, MASE，再根据各个误差指标的阈值检测出异常点，这里使用的模型为ExtremeLowDensityModel。

**2 时间序列预测模块基础模型解析：**

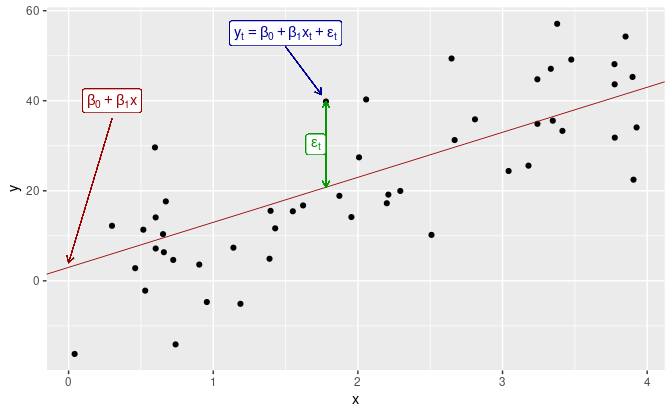
2.1 Naive预测法（NaiveForecastingModel）：

在Naive预测方法中，我们简单地将所有预测值设为最后一次的观测值，这种方法在很多经济和金融时间序列预测中表现得非常好，当数据服从随即游动过程时Naive方法是最优的。

2.2 简单线性回归（RegressionModel）：

最简单的线性回归模型假设被预测变量y和单个预测变量x之间存在如下线性关系：

其中，系数β0和β1分别表示回归线的截距和斜率。截距项β0表示当x=0时y的预测值；斜率β1表示当x增加一个单位时，y的平均变化。



从上图可以看出，观测值并不全部落在回归线上，而是分布在回归线的周围。我们可以这样理解：每个观测值 yt 都包含可解释部分 β0+β1xt 和随机误差项 εt。随机误差项并不意味着错误，而是指观测值与线性模型的偏差。它捕捉到了除 xt 外其他影响 yt 的信息。

2.3 多元线性回归：

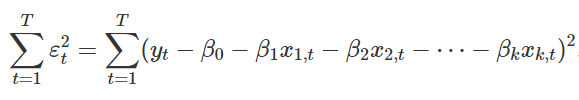
当预测变量有两个甚至更多时，模型被称为****多元线性回归模型****。多元线性回归模型的一般形式如下：yt=β0+β1x1,t+β2x2,t+⋯+βkxk,t+εt,

其中，y是被预测变量，x1,…,xk是k个预测变量，每个预测变量都必须为数值型变量。系数β1,…,βk分别衡量了在保持其他所有预测变量不变的情况下，该预测变量对被预测变量的影响程度。因此，系数衡量了对应预测变量对被预测变量的边际影响。

2.4 最小二乘估计：

在实际问题中，我们有一系列的观察值，但是我们不知道模型系数β0,β1,…,βk 的具体值。因此，我们需要利用模型对这些参数进行估计。

最小二乘估计方法通过最小化残差平方和来确定模型的各个参数。也就是说，我们通过最小化下式来确定 β0,β1,…,βk 的估计值：



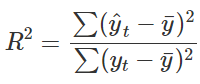
由于它的目标是最小化残差平方和，因此被称为****最小二乘估计****。寻找最优参数的过程，一般被称为“拟合”模型，或者被称为模型的“学习”或者“训练”。参数的估计值，一般用 β0,…,βk的预测值 来表示。

我们可以利用回归方程中的估计系数并将误差项设置为零来预测 y 。我们通常将模型写成如下形式：



将训练样本中x1,t,…,xk,t （其中 t=1,…,T ）的值代入模型中，我们将会得到 yt 的预测值，即为模型的*拟合值*。需要注意的是，这是模型估计得到的训练样本的预测值，而不是 y 未来真实值的预测值。

一般用可决系数（R2）评价线性回归模型对数据的拟合程度。它可以通过计算观测值 y 和预测值 y 之间的相关性来得出。或者，通过下式计算：



可决系数反映了回归模型所能解释的被预测变量的变异占被预测变量总变异的比例。

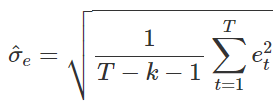
在简单线性回归模型中，R2也等于y和x的相关系数的平方（假设存在截距项）。

预测值越接近于真实值，R2则会越接近于1。相反，若预测值和真实值不相关，则R2=0（假设存在截距项）。在其它情况下，R2的值则会处在0和1之间。

但是仅仅利用R2来衡量模型是远远不够的。因为当增加解释变量的个数时，R2值将会不断增加，但这并不意味着更好的模型效果。

目前并不存在衡量R2值好坏的规则，R2值的有效性需要视具体情况而定。因此，利用模型在测试集上的预测结果来衡量模型好坏比直接根据R2大小来衡量模型更加有效。

另外一个衡量模型拟合效果的指标是残差的标准偏差，通常称之为“残差标准误差”。



其中，k是模型中预测变量的个数。需要注意的是，由于需要估计的参数个数为k+1（截距项和k个解释变量），因此上式中分母为T−k−1。

模型的标准误差和平均误差有一定联系。我们可以将标准误与y的均值或标准差做对比，得到一些关于模型精度的结论。

在生成被预测变量的预测区间时，标准误差将十分有用。

2.5 简单的指数平滑：

最简单的指数平滑方法自然被称为“简单指数平滑”。这种方法适用于预测没有明显趋势或季节因素的数据。

预测值使用加权平均值进行计算，其中权重随观测时间的久远程度呈指数型下降 — 最早的观察值被赋予最小的权值：

其中0≤α≤1是平滑参数。向前一步T+1时刻的预测值是时间系列y1,…,yT中所有观察值的加权平均值。权重下降的速度由参数α控制。

对于0到1之间的任何α值，随着时间向前推移，观察值的权重呈指数型下降，因此我们称之为“指数平滑”。如果α很小（即接近于0），那么更远的过去的观测值会被赋予更多的权重。如果α很大（即接近1），则赋予更多的权重给最近的观测值。 对于y^T+1|T=yT的极端情况，指数平滑的预测值等于朴素预测值。

2.6 霍尔特线性趋势法

Holt将简单指数平滑法扩展到可以预测具有趋势的数据。该方法包含一个预测方程和两个平滑方程（一个用于水平，一个用于趋势）：

预测方程：

水平预测方程：

趋势预测方程：

其中ℓt表示在t时刻该时间序列的水平的估计值，bt表示该时间序列在t时刻的趋势（斜率）的估计，α是水平0≤α≤10≤α≤1的平滑参数，β∗是趋势0≤β∗≤10≤β∗≤1的平滑参数。

与简单的指数平滑一样，这里的水平方程表明 ℓt 是观测值 yt 和 t 时刻的向前一步训练预测值的加权平均值，这里由 ℓt−1+bt−1 给出。趋势方程表明，bt 是基于 ℓt−ℓt−1 和前一个趋势的估计值 bt−1 在 t 时刻的估计值的加权平均值。

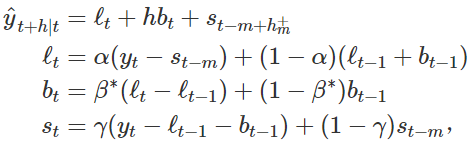
这里的预测函数不再平坦，而是有趋势的。向前 h 步预测值等于上一次估计的水平值加上前一个估计的趋势值的 h 倍。因此，预测值是一个关于 h 的线性函数。

2.7 Holt-Winters 季节性方法

Holt和Winters 将Holt方法进行拓展用来捕获季节因素。Holt-Winters季节性方法包括预测方程和三个平滑方程—一个用于水平ℓt，一个用于趋势bt，另一个用于季节性分量st，相应的平滑参数分别为α，β∗和γ。我们用m来表示季节频率，即一年中包含的季节数。例如，季度数据的m=4，月度数据的m=12。

Holt-Winters加法模型：

加法模型的分量形式是：



这种方法有两种不同的季节性组成部分。当季节变化在该时间序列中大致保持不变时，通常选择加法模型；而当季节变化与时间序列的水平成比例变化时，通常选择乘法模型。在加法模型中，季节性分量在观测序列的尺度上以绝对值表示，在水平方程中，时间序列通过减去季节分量进行季节性调整，并且每年的季节性分量加起来大约为零。在乘法模型中，季节性分量用相对数（百分比）表示，时间序列通过除以季节性分量来进行季节性调整，并且每年的季节性分量加起来约为m。

1. **基于传统模块的改进和完善：**
   1. LSTM模型：

（1）数据准备、预处理

神经网络具有很强的学习能力（也很容易过拟合），更适用于大规模数据集，因此需要准备大量、高质量并且带有干净标签的数据。从激活函数（sigmoid、tanh）可以看出，模型的输出绝对值一般在 0~1 之间，因此需要对数据进行归一化处理。常见的方法有：

1、Min-Max Normalization： x′=x−xminxmax−xmin

2、Average Normalization： x′=x−uxmax−xmin

3、log function： x′=log10x

1、2 属于线性归一化，缺点是当有新数据加入时，可能导致 max 和 min 的变化，需要重新定义。3 属于非线性归一化，经常用在数据分化比较大的场景，有些数值很大，有些很小。

与归一化相近的概念是标准化：

Z-score规范化： x′=x−uσ

什么时候用归一化？什么时候用标准化？

1、如果对输出结果范围有要求，用归一化。

2、如果数据较为稳定，不存在极端的最大最小值，用归一化。

3、如果数据存在异常值和较多噪音，用标准化，可以间接通过中心化避免异常值和极端值的影响。

（2）批处理

神经网络一般不会一个一个的训练样本，通常采用 minibatch 的方法一次训练一批数据，但不要使用过大的批处理，因为有可能导致低效和过拟合。

（3）梯度归一化、梯度剪裁

因为采用了批处理，因此计算出来梯度之后，要除以 minibatch 的数量。如果训练 RNN 或者LSTM，务必保证 gradient 的 norm 被约束在 5、10、15（前提还是要先归一化gradient），这一点在 RNN 和 LSTM 中很重要。在训练过程中，最好可以检查下梯度。

（4）学习率

学习率是一个非常重要的参数，学习率太大将会导致训练过程非常不稳定甚至失败。太小将影响训练速度，通常设置为 0.1~0.001。

（5）权值初始化

初始化参数对结果的影响至关重要，常见的初始化方法包括：

1、常量初始化：把权值或者偏置初始化为一个自定义的常数。

2、高斯分布初始化：需要给定高斯函数的均值与标准差。

3、xavier 初始化：对于均值为 0，方差为（1 / 输入的个数） 的均匀分布，如果我们更注重前向传播，可以选择 fan\_in，即正向传播的输入个数；如果更注重反向传播，可以选择 fan\_out, 因为在反向传播的时候，fan\_out 就是神经元的输入个数；如果两者都考虑，就选 average = (fan\_in + fan\_out) /2。对于 Relu 激活函数，xavier 初始化很适合。

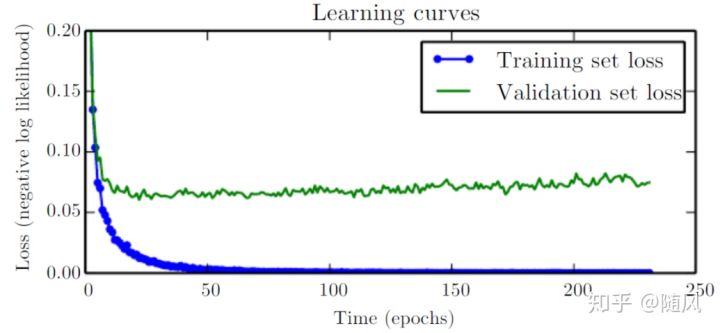
在权值初始化的时候，可以多尝试几种方法。此外，如果使用 LSTM 来解决长时依赖的问题，遗忘门初始化 bias 的时候要大一点（大于 1.0）。

（6）dropout

dropout 通过在训练的时候屏蔽部分神经元的方式，使网络最终具有较好的效果，相比于普通训练，需要花费更多的时间。记得在测试的时候关闭 dropout。LSTM 的 dropout 只出现在同一时刻多层隐层之间，不会出现在不同时刻之间（如果 dropout 跨越不同时刻，将导致随时间传递的状态信息丢失）。

（7）提前终止

在训练的过程中，通常训练误差随着时间推移逐渐减小，而验证误差先减小后增大。期望的训练效果是：在训练集和验证集的效果都很好。训练集和验证集的 loss 都在下降，并且差不多在同个地方稳定下来。采用提前终止的方法，可以有效防止过拟合。



训练集、验证集误差

1. **异常检测模块使用方法以及接口介绍：**
   1. 数据获取：

从设备端获取原始数据，暂定接口为：

<http://192.168.100.100:8080/messages?measurement=/SenseHatB/socc3dzf83ty/Temperature&period=7200>

获取两个小时的数据用于异常检测。

* 1. 数据处理：

APIInputProcessor.java程序用来将原始数据处理成TimeSeries形式并进行下一步处理。

4.3：数据训练：

DetectAnomalyProcessable.java的process方法用来使用配置文件中指定的模型生成的ModelAdapter的train方法训练数据。

4.4：异常检测：

DetectAnomalyProcessable.java中先创建AnomalyDetector模型，再用这种模型的detec方法进行异常检测获取到异常数据序列。

4.5：SmartTrainModel特殊模型详细解析：

这一模块是调用了python写的LSTM模型来训练数据，python服务器启动之后，SmartTrainModel组织好数据之后使用post方法向其传参，python端训练完数据之后返回我们的异常检测模块。由于这一模块比较独立，且和原本的代码基本无冲突，所以在保持原本代码封装完好的情况下，将其也融入了AutoForecastModel。