**内容概述：**

1. 利用甲烷存储COFs数据库微调MOFTransformer模型；
2. 完成了针对CoRE-COF的氧气吸附计算
3. 得到氧气吸附属性预测器

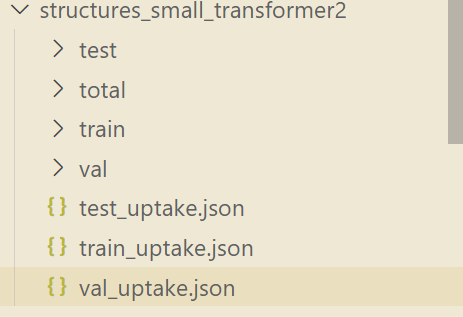
**具体内容：**

1. **利用甲烷存储COFs数据库微调MOFTransformer模型**

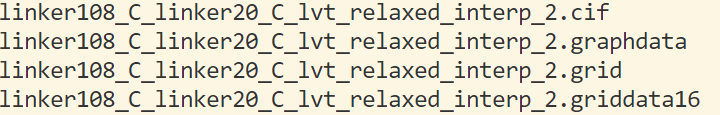
**任务内容：**第一步，将原始的cif文件转换为MOFT所需要的格式；第二步，利用这些数据微调MOFT得到高压甲烷吸附量预测器。

**完成情况：**第一步已经完成，数据已经转换为了MOFT所需要的格式；第二步开始进行。

**转换后的格式分析：**原始的cif文件一共有**69748**个，转换完成后在存储在一个新的文件夹下，目录格式如下：



其中，total目录下保存了所有成功转换的COFs，对于每一个COF，原始的cif文件将会被转换为四种文件——原本的cif文件、图结构信息文件、两种能量晶格文件，如图：



Train/val/test是对total进行划分后的数据，这里，我采用的划分方式是随机划分**8:1:1。**

Train\_uptake.json/val\_uptake.json/test\_uptake.json对应这属性文件，属性名是uptake。

**注意：因为数据转换只需要进行一次转换，对于后序还需要做的低压甲烷吸附量预测，只需要添加三个属性文件就可以了，分别命名为train\_low.json/val\_low.json/test\_high.json来区分。**

**微调过程：**MOFT提供了微调的代码，只需要修改一下目标地址就可以了



训练100个epoch，训练完成后再测试一下结果。

1. **完成了针对CoRE-COF的氧气吸附计算**

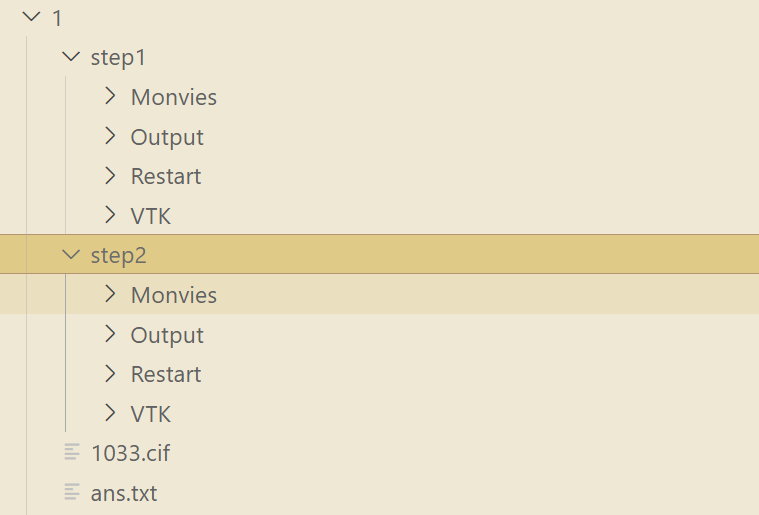
**任务描述：**见周报9-10

**具体方案：**见周报9-10

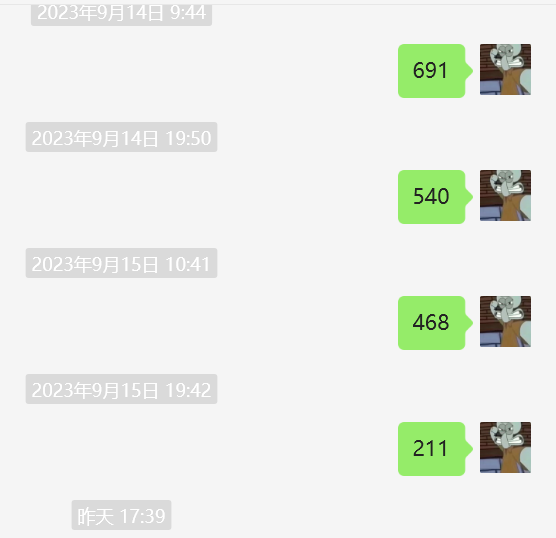
**问题以及解决方案：**

**问题1：**部分COF计算出来的结果是0，具体见周报9-10描述。

**解决方案：**gcmc计算分为两步，第一步是计算孔隙率，第二步才是计算氧气吸附，这两步的输出文件的格式是相同的。由于原先的代码在查找结果的时候没有区分两步的输出文件，所以在查找第二步氧气吸附值的时候会检查到第一步的输出文件，而第一步中没有氧气吸附属性，导致最终返回的结果是0。为了解决这个，我将每一步的结果分别利用一个单独的文件夹存储，将其区分开。



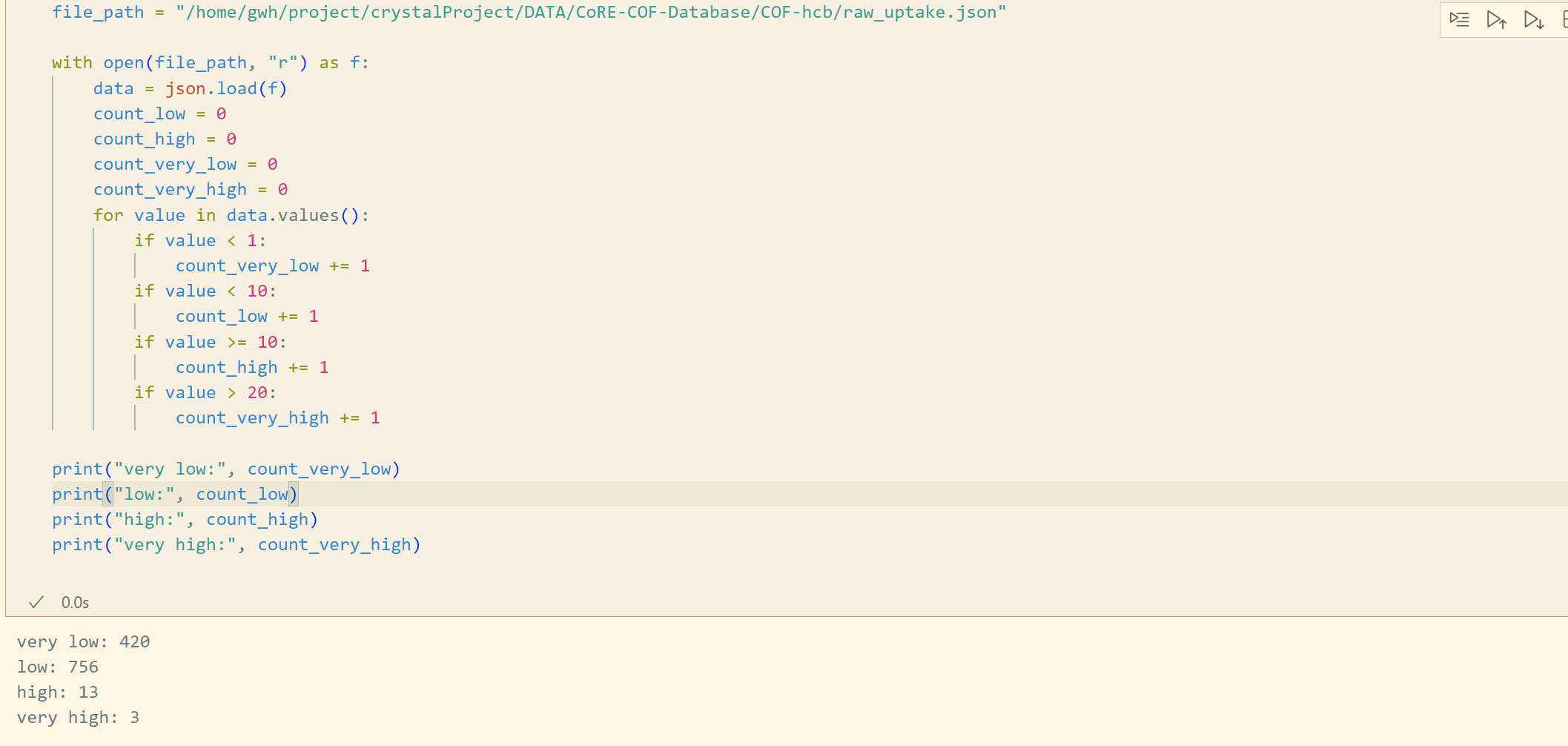
**问题2：**部分COF计算速度非常慢，并且容易程序崩溃，算的过程中程序突然退出，并且没有任何报错信息。但是数量非常少，只有



691.cif、540.cif、468.cif和211.cif这四个文件。

**解决方案：**为了尽快得到氧气吸附预测器，目前暂时放弃这些COF，跳过他们继续计算。

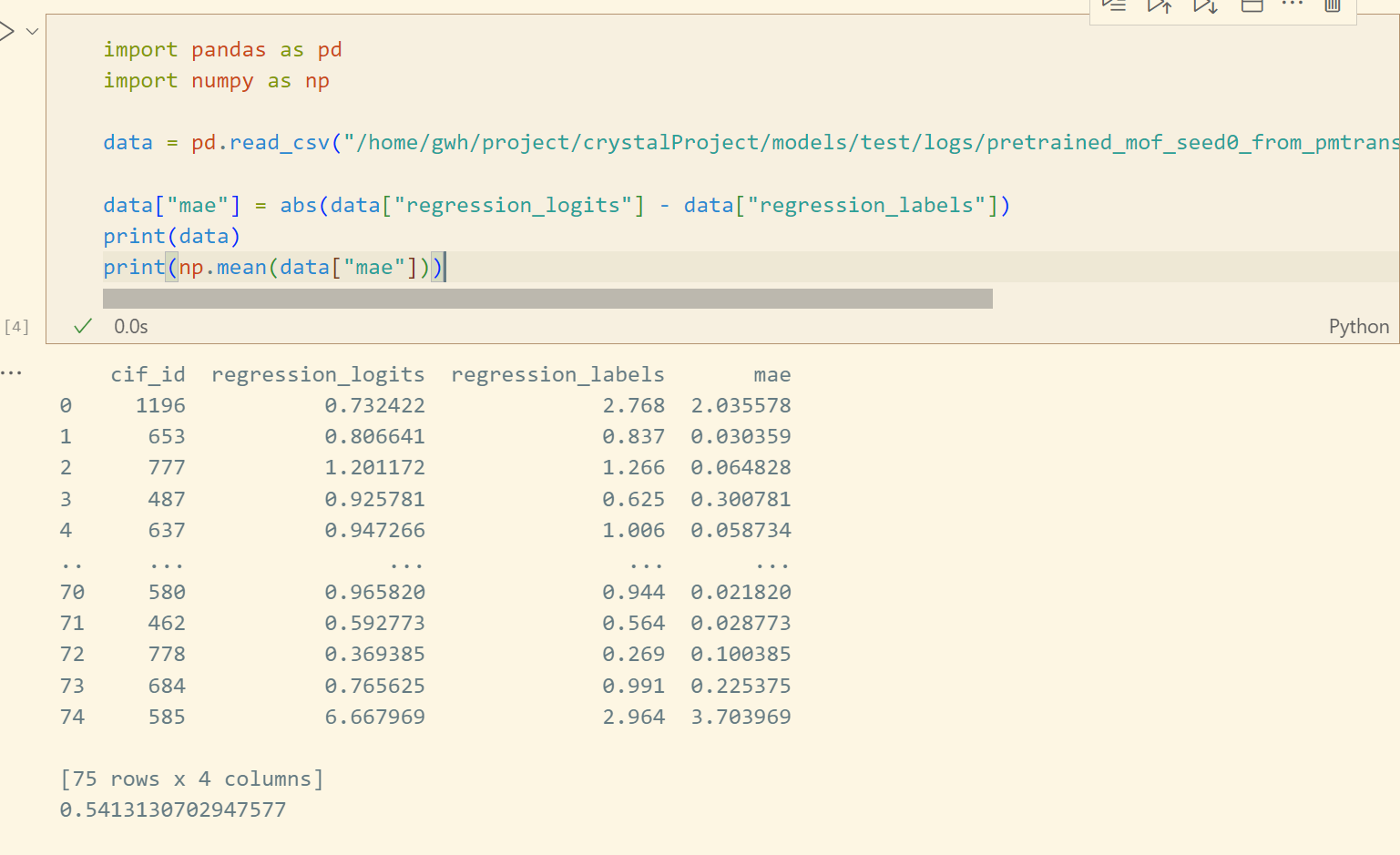
**结果分析：最终一共计算了769个COF,通过观察，大部分的COF的吸附属性都非常小，只有少数几个的吸附属性较好。**



1. **得到了氧气吸附属性预测器**

**任务描述：**利用第2个任务中已经计算好的氧气吸附属性，训练了一个氧气吸附属性预测器，训练过程采用随机的划分8:1:1

**效果分析：**统计了一下在测试集上的预测效果。



测试集上一共75个属性，mae是0.54，作为第一步的预测器，效果还可以了，并且对于mae较大的原因是由于其中某几个COF的预测误差较大，大部分COFs的预测误差都较小。

**第二版预测器：**

**代码已经全部写好，上传到了gitlab上，准备下周运行rl，测试效果。**

**下周安排：**

1. **利用甲烷存储的COFs数据微调MOFTransformer模型**
   1. **数据格式的转化已经全部完成了**
   2. **下周开始训练模型**
2. **开始写拓扑深度学习的代码**
   1. **搞清楚了高阶网络的消息传递范式，知道了代码应该怎么写**