2025年7月9日

因两次灾难性bug而严重迟滞

下一步指示

那么，接下来我该focus哪个方向?
0. 你把压缩以后的平均长度和entropy对比一下
    -- [已经支持对发生负优化的bucket进行分析，但是对于目前所有的bucket还暂无支持]
0. 你说的压缩以后反而变大的case对应的分布的entropy和shape是什么?
    -- [shape部分发生了overflow错误，无法收敛，仍需修复。但是entropy计算完毕，两者似乎没有关系，详见附上的scatter plot]
1. 修好pt2h5的逻辑
    -- [未完成]
2. 更换模型为gemma或者其他模型(您建议是?)
    -- [未完成]
3. 尝试对tensor进行分桶
    -- [已完成，详见今天的安排]

4. 尝试模仿pytorch的 bf16 compression hook，将EG Compression加入训练流程? -- [0,1,2,3完成后再做]

今天的安排：

1. 对昨天的数据进行分析:
    正在分析列: ['ratio_os']
    ```
              ratio_os
    count  10084.000000
    mean       0.474154
    std        0.151435
    min       -0.855337
    25%        0.407508
    50%        0.509289
    75%        0.554271
    max        0.872622
    ```
    ![2025年7月9日_1](img/25_7_9_1..png)

    是的，绝大多数都可以很好的压缩，但是还是有几个别出现了反向优化的情况，更有甚者甚至达到了反向85%的优化。
    列 ratio_os 中负值的数量: 106 —— 也就是有约1.05%的bucket数显了反向优化

    我现在正在研究一个cal_shape_entropy以期计算出他们的情况

2. 完成了对于tensor进行bucket化的函数，添加进了cal_compression中，名为bucketize.py
    逻辑如下：
    bucketize(unbucketized_tensor:torch.tensor,bucketize_scale:int = 100) -> bucketized_tensor: torch.tensor

    读入一个tensor，然后：
    按照bucketize_scale (默认为100，可以根据实际情况调整)从小到大分成共{len(unbucketized_tensor)/bucketize_scale}个桶
    桶的值为："(桶的理论最大值 + 桶里理论最小值) / 2"
    将tensor中的每一个entry放进某一个桶里

3. cal_entropy_and_shape:
    通报：
    matlab和python的调配发生了灾难性的bug，尝试修复无果。
    已更换纯python实现计算相应数据

    结果：
    对全部106条数据(总数据约1.05%)进行计算，发现entropy似乎和发生负优化的情况关系不大？

    请看这些散点图：

    ![2025年7月9日_2](img/25_7_9_2..png)
    ![2025年7月9日_3](img/25_7_9_3.png)

    shape因为发生了灾难性的overflow，仍在抢修……