jetson部署sgmnet研究报告

研究内容

在jetson AGX orin上面部署sgmnet，尝试了以下四种方法：

torch2trt

由于网络结构过于复杂，最后无法转换成功。

转onnx推理

这是一个比较经典的办法，但是输入必须是numpy数组，从CPU上面将数据传输到GPU里面开销较大，最后只能达到0.8s每帧的速度。

多线程

这个方法在写C++的时候是可以用的，但是工程量太大，在python中开启多线程的时候，实测速度没有提升，后面去查找发现python是伪多线程，是将一个线程拆分成多个线程，无法调用CPU的多个核心，这个问题在python4出来过后将得到解决

torch2onnx2trt

该方法是目前四个尝试中消耗工作量最大的方法，首先是在转换为onnx的过程中，出现了警告，显示网络被设置成了训练模式，但实际上在代码中已经使用过 `model.eval()`

而且去一一寻找，将所有的test_mod都设置成了True，但是警告仍然没有消除，接下来是在onnx2trt的过程中，tensorrt8发生了报错，原因是因为不支持转topk算子，类似的算子不支持问题还有很多，错误日志一大段，后面换成了tensorrt10，虽然在转换trt时成功通过，但是在推理的时候，出现了数据进入模型过后，遇到不支持的句柄弹出的情况，原因有很多，目前发现的问题是torch.topk的输入中有一个参数必须是int64，但是tensorrt不支持int64这个数据类型，且该参数是从配置文件中读取，此矛盾问题无法解决，后期工作无法开展。

研究结论

上述尝试过的方法，在实践中发现很难达到要求，我们同时还可以对AGX和4090进行规格参数上面的简单计算，验证可行性，其Cuda核心只有4090的一半，核心频率少了接近1G，唯一多出来的计算资源只有2个DLA，但是DLA主要用于加速卷积计算，也就是resnet这一类的网络可以有比较好的效果，但是对于使用了注意力机制的网络，里面大量的算子都无法进行加速。在数据全部加载到cuda的情况下，AGX速度也只能达到0.5s，而4090只需要0.06s，在加上了网络的前后处理过后，即便是在4090上，也只有6帧左右的速度，从纯硬件方面进行估算，想达到10帧每秒，是非常困难的

未来建议

对于一个算法的量化部署，涉及到从软件pytorch到硬件的优化，需要有如下注意事项，首先排除各种自定义的量化方法，因为二值化等量化方法，在真实的cuda设备上面部署时，仍然是按照fp16或者fp32进行储存的，并没有加速效果，二值化等方法的加速效果只存在于高度自定义的硬件上面，其次是构建网络的时候，注意计算图的数据类型，尽量都是torch的tensor，不要跟python原本的数据格式混用，最后是算子的使用，当一个工作已经考虑到后期需要进行量化的时候，就要在一开始注意算子对应，详细可以参考链接中的表格

https://github.com/onnx/onnx-tensorrt/blob/84b5be1d6fc03564f2c0dba85a2ee75bad242c2e/operators.md

总而言之，量化工作不同于在服务器上训练推理网络，在设计之初就要考虑网络的可部署性，遵循对应规则，不然后期会浪费大量时间在解决冲突上面。