2023秋 分布式机器学习系统 warmup

基本信息

1. 作业截止时间：10月9日 23:59 (no grace day)
2. 提交方式：通过教学网提交
3. 提交内容：实验报告和源代码，要求见“具体任务”和“评分标准”

作业目的

1. 熟悉 DNN 训练、测试环境及流程（本作业使用PyTorch）
2. 理解 DNN 框架中的算子在 GPU 上的加速原理
3. 熟悉 CUDA 实现和优化算子的过程

具体任务

1. 理解 PyTorch 中 LayerNorm 算子的计算过程，推导计算公式和反向传播公式
2. 了解 GPU 端加速的原理，CUDA 内核编程和实现一个 kernel 的原理
3. 实现 CUDA 版本的定制化 LayerNorm 算子的前向传播
    a. 在.cu 文件中，编写使用cuda 进行前向计算的函数，参数和输出格式参考 PyTorch LayerNorm （https://pytorch.org/docs/1.5.0/nn.html#layernorm)。注意：只需实现 normlized_shape 为最后一维 dim size 且 elementwize_affine=False 的情况。
    b. 基于 C++ API，编写.cpp 文件，调用上述函数，实现 LayerNorm 算子的前向计算
    c. 将代码生成 python 的 C++扩展，并使用基于 C++的函数扩展，实现自定义 LayerNorm 类模版的前向计算函数
    d. 运行程序，验证算子的正确性和测试算子执行速度（取 100 次执行的平均值），验证测试参考<code>/custom_linear.py
4. 下述任务为选做任务，但要求至少选择其中一项任务，多选加分
    a. 类似前向传播，使用 CUDA 实现 LinearNorm 算子的反向传播，编写.cpp 文件调用反向传播，并生成python 的 C++扩展，完成自定义 LayerNorm 类模版，运行程序并验证算子正确性和测量算子执行速度（取 100 次执行的平均值）
    b. 使用 nvidia 和 pytorch 提供的 profiler（如 nvprof、nsight compute、nsight systems、 pytorch profiler 等），分析自定义算子性能、资源使用和其他特征，比较 CUDA 对算子性能的影响。

评分标准

1. 本次作业需按时提交实验报告和源代码
    a. 实验报告应包含：实验环境（软硬件环境等）、实验结果（简要的实现描述、正确性比较、性能比较、选做任务等）、总结与体会等；
    b. 源代码应包含可复现实验结果的所有代码、执行脚本和简要的 README，以任何人可以快速复现实验结果为标准。
2. 本次作业分数 G=min(100，基本分数+性能分数+加分)；
3. 正确性：实验报告中应包含自定义算子与pytorch 算子执行结果的比较，将依据正确性给出本次作业的基本分数（满分 60）；
   注意：np.testing.assert_allclose()参数设置为rtol=1e-3，atol=1e-5，未assert即为通过；将仅设置rtol=1e-3即可通过计入选做任务加分；
4. 性能：实验报告中应包含自定义算子 100 次执行平均的执行速度，将依据执行速度给出本次作业的性能分数（满分 30）；
5. 加分：根据可选任务 4 的实现情况，适当加分（满分 10 分，最多加20分）。

实验环境

1. 以小组为单位统一提供未名一号机器，组员需联系组长获取账号，分组详见：2023分布式课程资源分组
2. 本次作业也可使用其他个人可用的GPU，但需标明GPU型号

参考资料

1. 参考自定义 Linear 算子的代码：<code>。运行环境：PyTorch1.5.0+CUDA10.1。运行命令：

```
cd mylinear_cuda_extension
python setup.py install --user

cd .. && python mnist_custom_linear_cuda.py
```

   来源：MSRA AI-System Lab3 https://github.com/microsoft/AI-System/tree/main/Labs/BasicLabs/Lab3
2. CUDA 编程
    a. https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html
    b. https://developer.nvidia.com/blog/even-easier-introduction-cuda/
3. pytorch 自定义算子
    a. https://pytorch.org/docs/1.5.0/notes/extending.html?highlight=custom%20c
4. 性能优化

a. OpenMP：https://www.openmp.org/

b. SIMD AVX：https://en.wikipedia.org/wiki/Advanced_Vector_Extensions

c. C++ thread：https://cplusplus.com/reference/thread/thread/

d. pthread：https://www.geeksforgeeks.org/thread-functions-in-c-c/

5. 未名一号文档

a. https://hpc.pku.edu.cn/_book/

b. 注意：因未名一号 login 和计算节点无法联网，如需下载/传输数据或安装软件，需在 data 节点进行