week9实验记录

zxp

November 25, 2023

1 environment

cpu:Inter i5-12400f (2.5 GHz)

System:Ubuntu 22.04.1

Compiler:gcc 11.4 和gcc 12.3

2 code

本周所使用的代码和上周完全一致，只改了编译选项和环境

3 Experiment

这部分使用的是gcc 12.3。上周发现-march=native这个选项没有达到预期,怀疑是编译器的问题(之前用的是gcc 11.4)，所以换了个编译器，本来想用最新的gcc 13.1。但是，我使用的CUDA最高支持gcc的版本为gcc 12，所以使用gcc 12.3重新绘制了在O2、O3、Ofast和Ofast -march=native直接卷积花费的时间和gflogs。

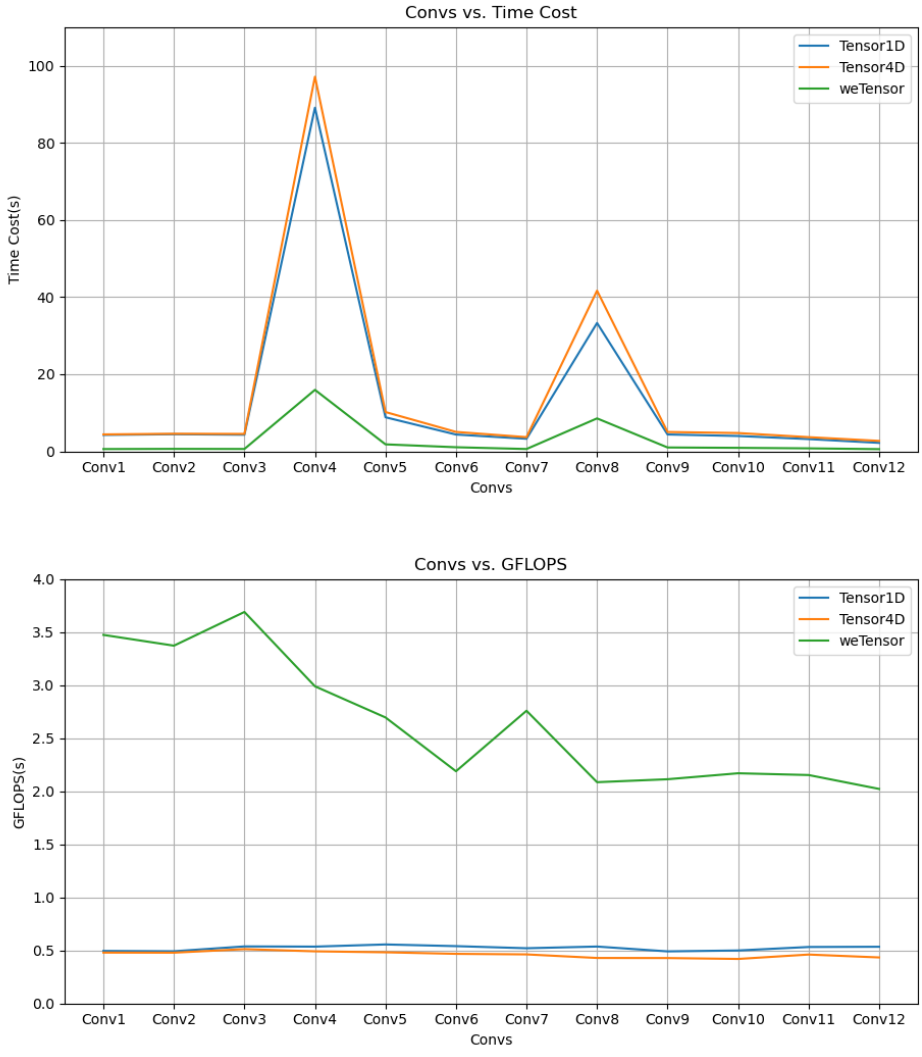

Figure 1: Compilation Optimization option: -O2

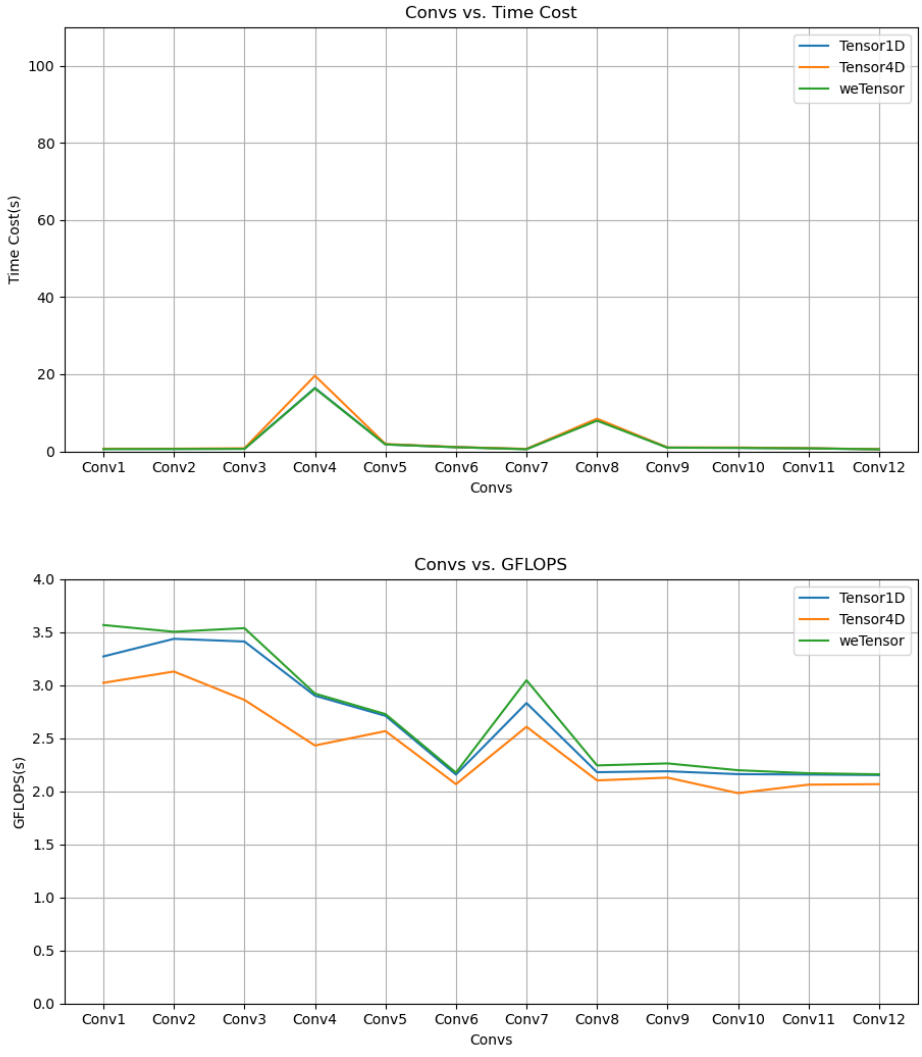

Figure 2: Compilation Optimization option: -o3

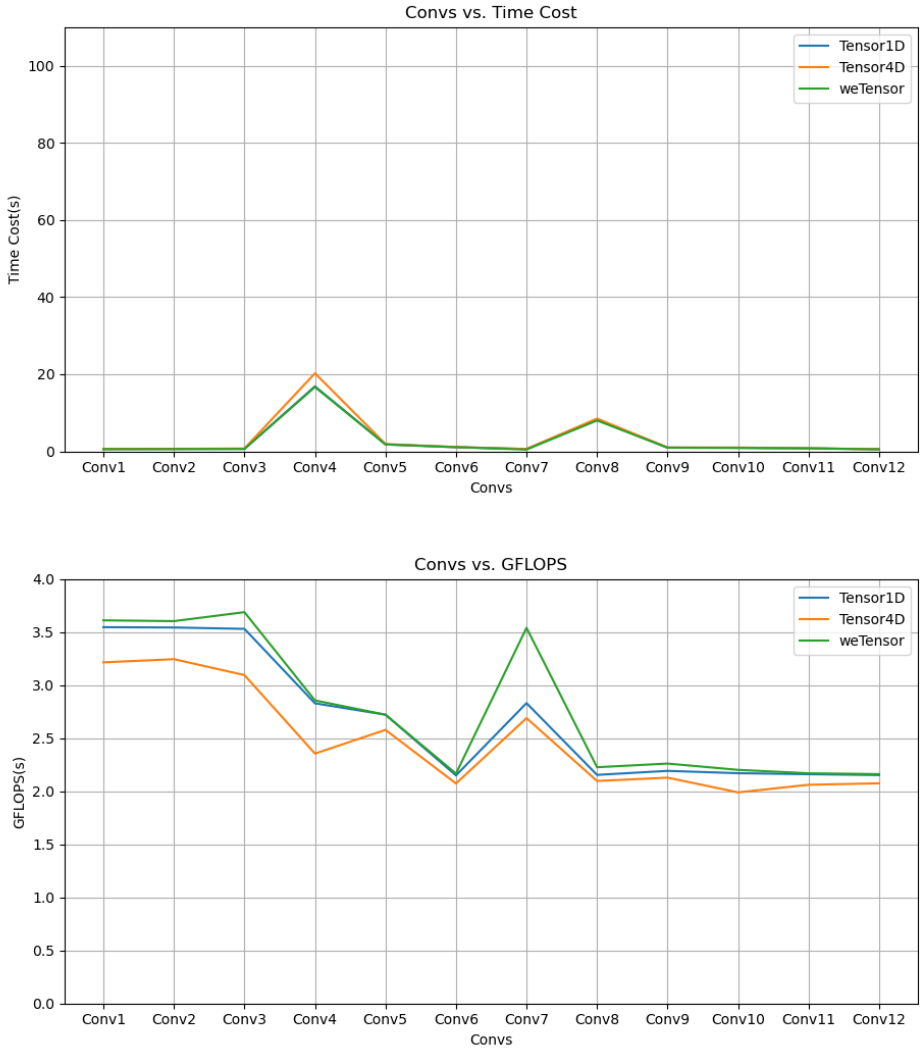

Figure 3: Compilation Optimization option: -Ofast

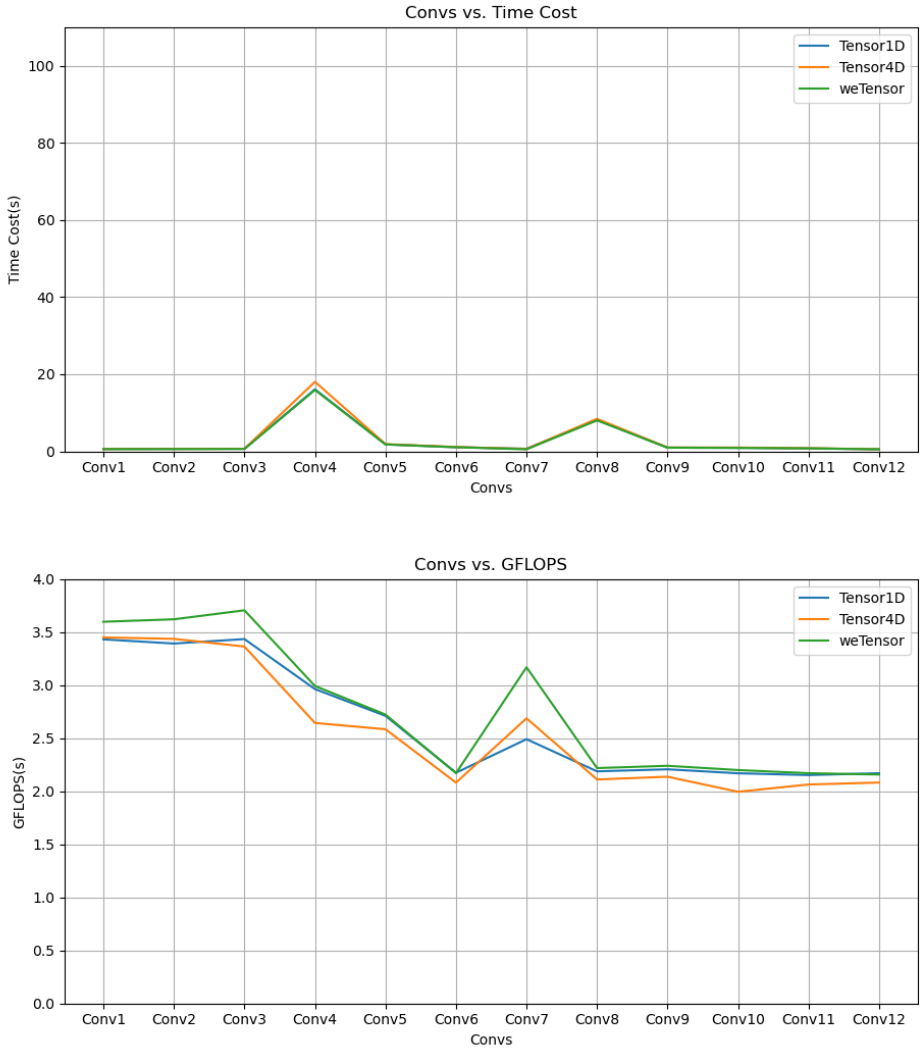

Figure 4: Compilation Optimization option: -Ofast -march=native

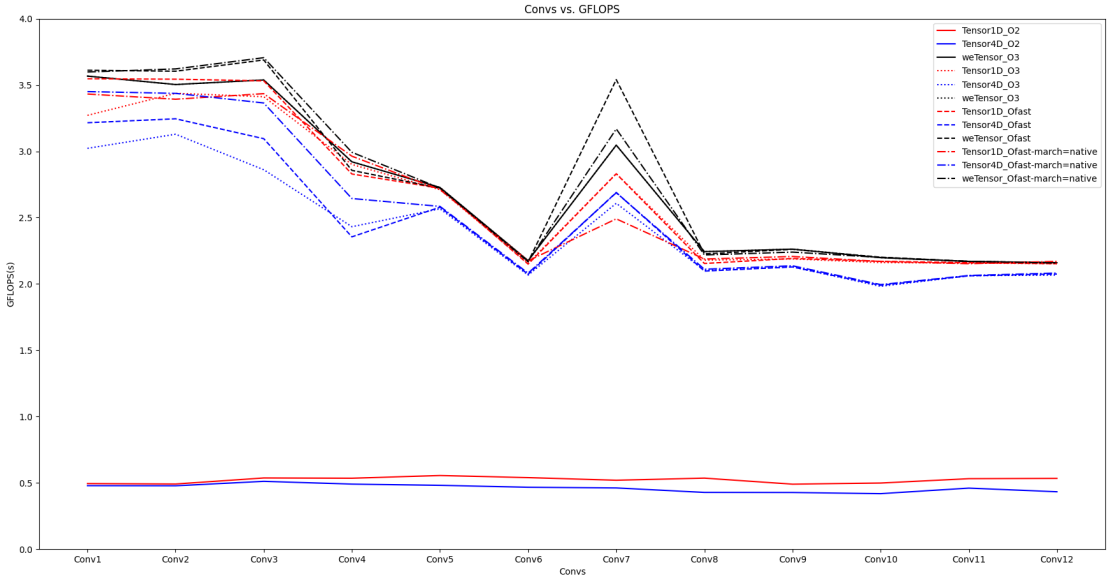

Figure 5: 把曲线一起比较

3.1 Analysis

使用gcc 12.3为编译器后，-march=native这个选项就不会像在gcc 11.4编译器那样大幅度降低性能，虽然在Conv7的情况下-march=native表现不是最好，但部分Conv（比如Conv1）下开了-march=native比不开表现好。

4 Experiment2

这周我在unbuntu原生环境下编译了libtroch的cpu版本（分别使用了openblas和mkl线性代数库）和官方编译的libtroch的GPU版本放一起进行比较（详细的libtroch信息看本文件旁边的txt文件）。
因为担心有误差，每个版本运行了八次，下面展示的是平均值。
这部分使用的编译器是gcc 11.4（跑的时候还没安装好gcc12，所以使用了gcc 11.4）
数据没有处理完，只画了O2的图，详细看本文件旁边的xlsx文件

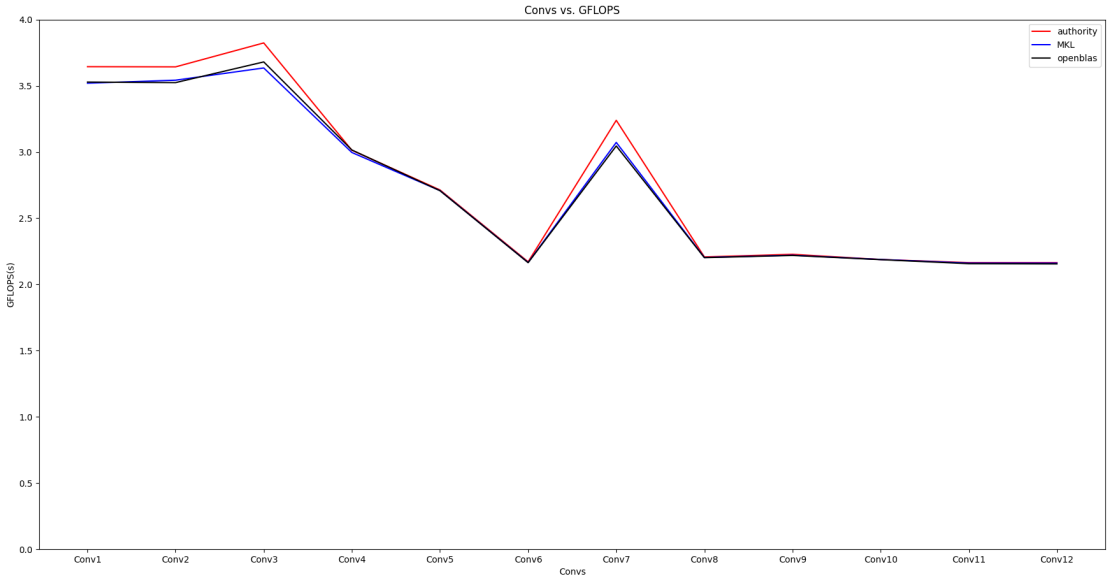

Figure 6: 把曲线一起比较

4.1 Analysis

MKL和openblas表现差不多，但官方编译的部分情况下快一点

5 Experiment3

查看汇编代码，详细看本文件旁边的txt文件。这部分使用的编译器是gcc 11.4。
-fipa-cp-clone这个优化是让调用函数的时候不传入常量，在调用函数传常量的时候拷贝函数实现一个不传常量的版本。我们的代码调用直接卷积函数传入的是带const，所以直接卷积函数被拷贝了，这样就和前几周写的直接卷积函数是在类里面情况接近了。但wetensor使用了libtorch里的数据结构，该数据结构不允许设置为const，所以理论wetensor是没享受到这个优化的。
因为反汇编得到的汇编代码不如用godblot网站上直观，调用的函数显示的函数名不是源码中的，我没具体找到wetensor函数，按着tensor1d/4d的特征粗略的对比了一下，wetensor部分没明显克隆函数
我用的objdump看汇编代码，还在找有没有更好的方式看汇编代码或更仔细的查看。

5.1 Analysis