Compare Tensor 2

Hao Deng

November 2023

1 Introduction

In the last week, we implemented Tensor using one-dimensional vector and four-dimensional vector, respectively. One drawback of our previous assignment was not using standard tensor dimensions in the industry. Therefore, we will focus on comparing two tensor implementations using twelve convolution layers of the DNN benchmarks in this assignment. We will use the following compilation optimization options:

1. -O2
2. -O3
3. -Ofast -march=native

Our machine has:

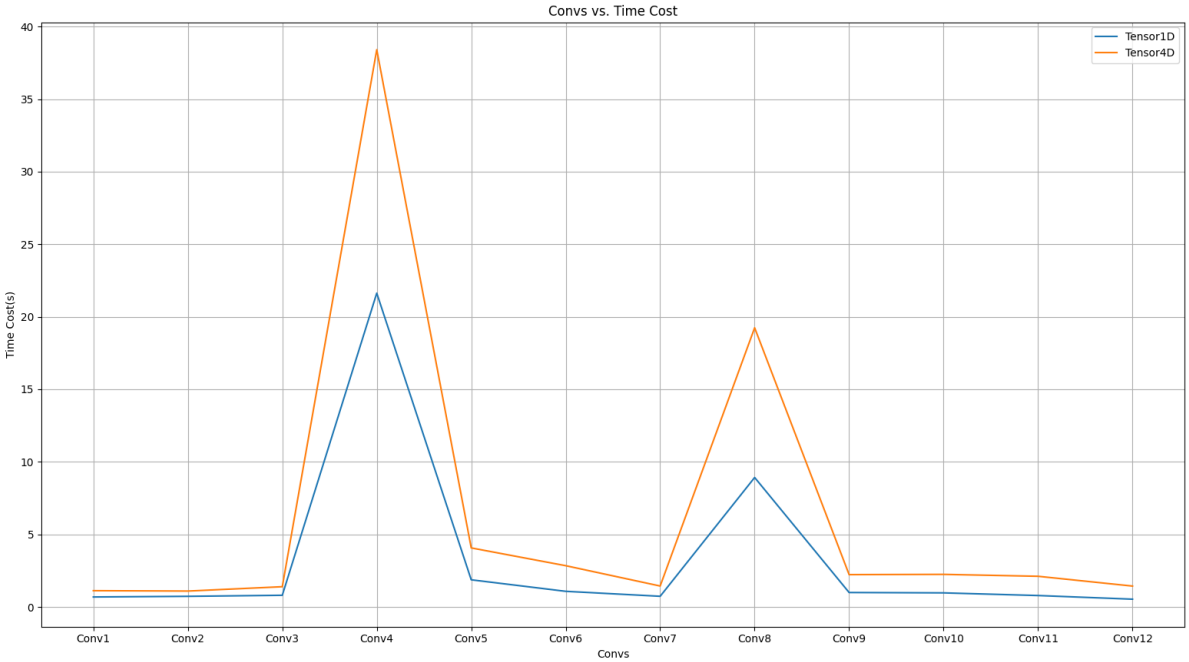
1. CPU: Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz 3.70 GHz
2. System: Windows 10
3. Compiler: gcc version 8.1.0 (x86_64-win32-seh-rev0, Built by MinGW-W64 project)

Project Repo:
https://github.com/SakakibaraMako/Benchmark

2 Experiment

Since the twelve convolution layers of the DNN benchmarks don't specify the batch sizes of the input tensors, we fix the batch size for input at 10.

We recorded the performance of twelve convolutions with Tensor1D and Tensor4D using different compilation optimization parameters and generated the following plot:



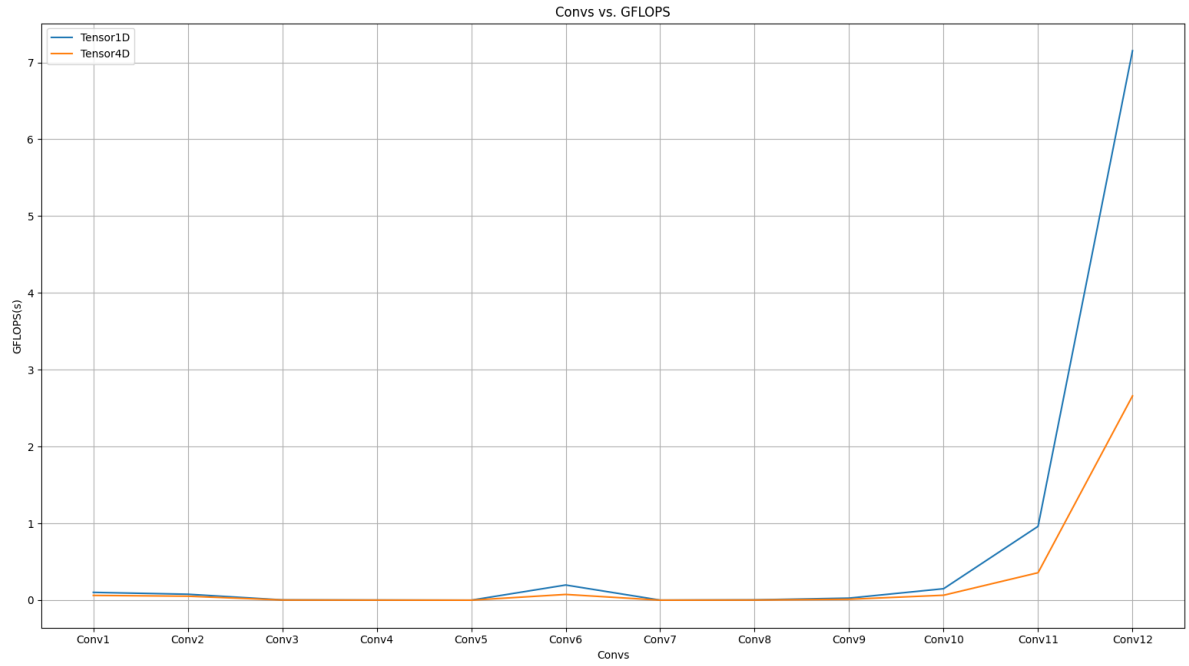
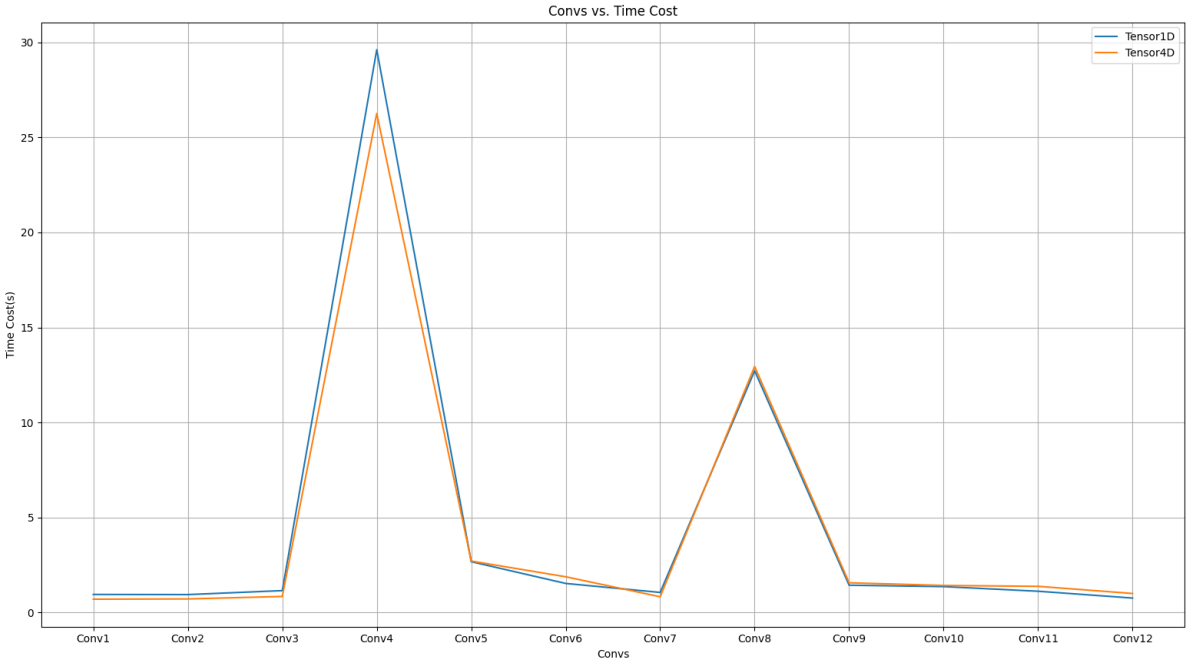


Figure 1: Compilation Optimization option: -O2



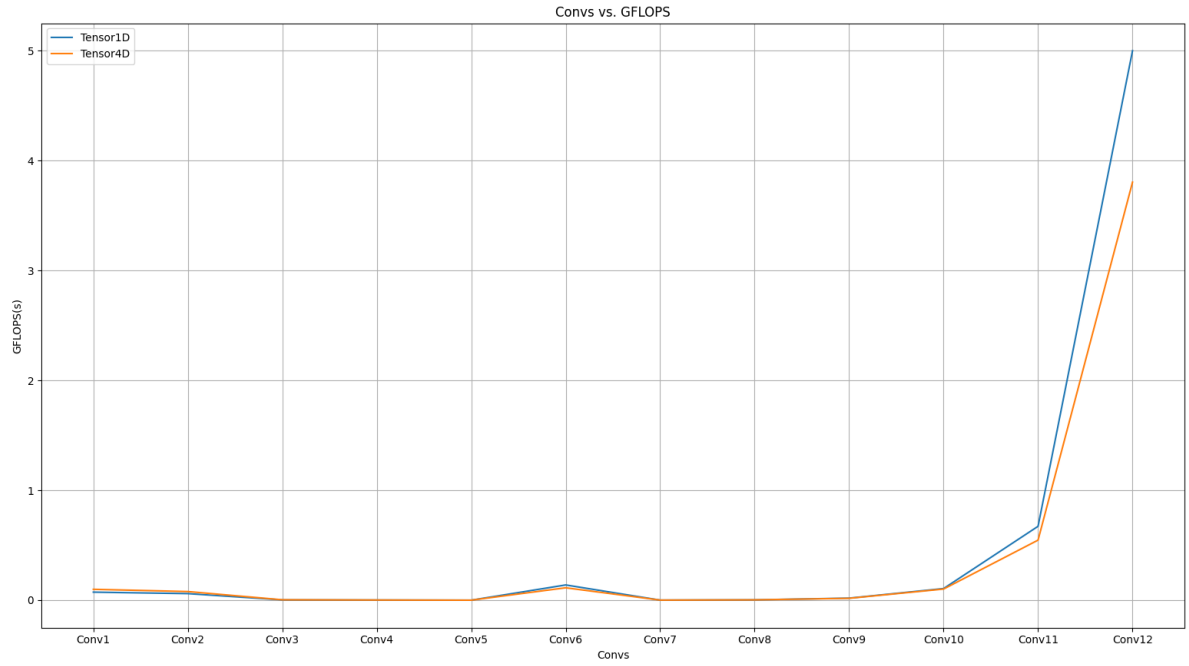
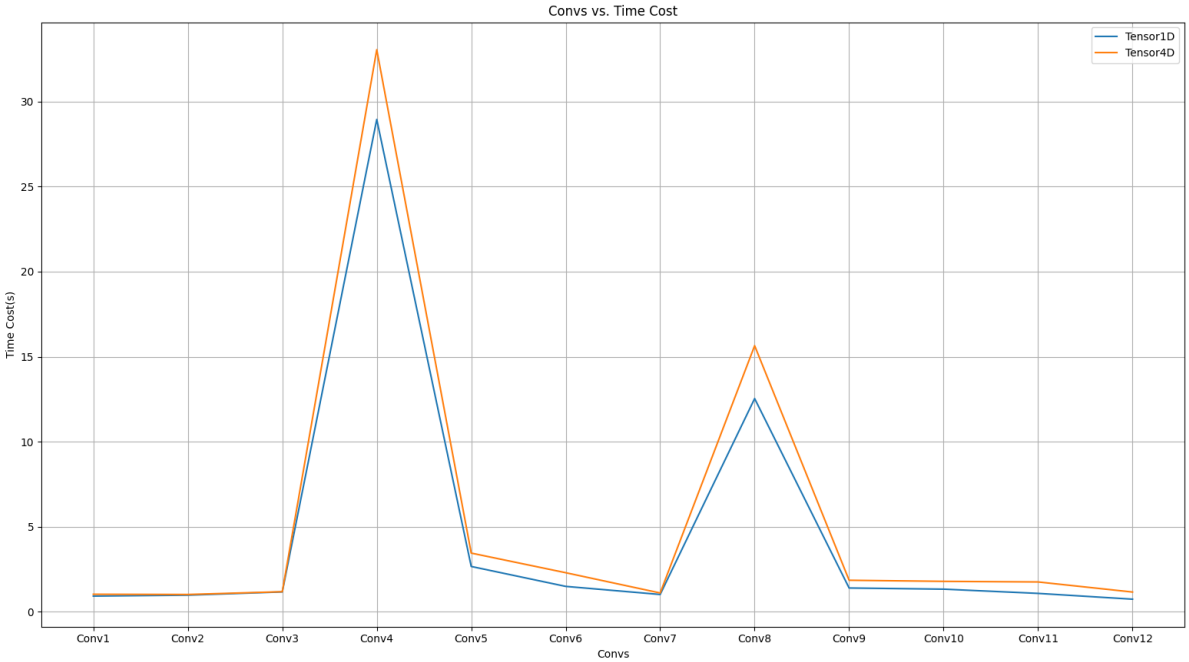


Figure 2: Compilation Optimization option: -O3



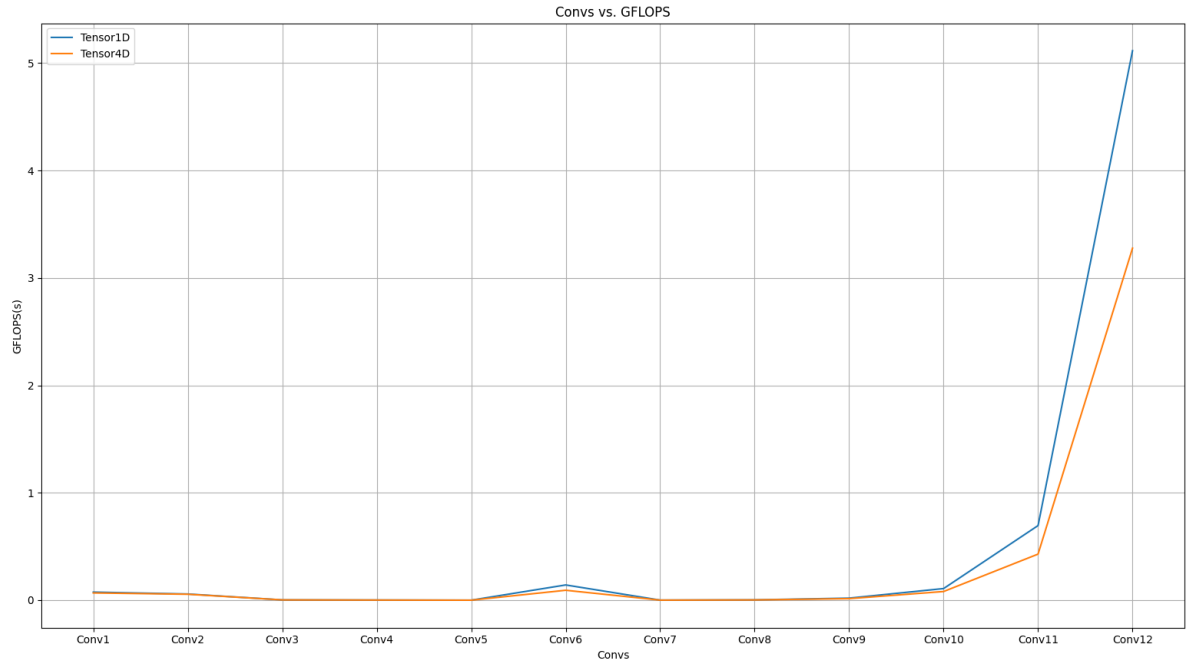


Figure 3: Compilation Optimization option: -Ofast

3 Analysis

We notice that under -O3 and -Ofast -march=native, there is little difference between the performance of Tensor1D and Tensor4D. However, Tensor1D is much better than Tensor4D in terms of runtime under -O2.

Another interesting pattern to notice is that Conv 4 and Conv 8 spent significantly more time than other layers while their GFLOPS are approximately equal to that of other layers. We speculate that this pattern was caused by significantly larger amount of data in Conv 4 and Conv 8.

Finally, we found that Conv 12 had extremely good performance in terms of GFLOPS. The second best is Conv 11, follow by Conv 6. These layers have one thing in common, which is small height and width. This led to better locality and hence, better performance.

4 Conclusion

Under -O2, Tensor1D is better than Tensor4D; under -O3 and -Ofast -march=native, there is little difference between Tensor1D and Tensor4D.