CSC2222H Project Proposal

Halide for accelerating depthwise separable convolution layer

Group name: EP/D Author: Runqing Zhang, Yihan Duan

Summary

Halide is a programming language designed for high performance image and array processing. Halide provides code auto-generation and optimization functionality but can also be hand-tuned for even better performance. In recent years, many have experimented with the use of Halide in accelerating deep learning training and inference. See ***Differentiable programming for image processing and deep learning in halide*** and ***Accelerate DNN Performance with Sparse Matrix Compression in Halide***.

In recent years, there has been constant effort to reduce the total number of parameters in deep neural networks for faster training and inference. A successful method that reduces parameters without compromising accuracy is using depthwise separable convolution layers to replace normal convolution layers. See ***Xception: Deep Learning with Depthwise Separable Convolutions*** and ***MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications***. The main idea of a depthwise separable architecture is to replace a normal convolution layer (N x N x C) with a channel-wise convolution layer (N x N x 1) and a point-wise convolution layer (1 x 1 xC). This method is fairly popular because it results in several magnitude fewer parameters while maintaining relatively high precision.

Problem

In the official Halide applications directory, both normal convolution layer and depthwise separable layer are implemented and optimized. However, unlike normal convolution layers which utilized around 90% peak Flops for different GPUs, depthwise separable convolution layers only used 1.2 TFlops (2.3%) on RTX2060, far from the peak 52 TFlops. On CPUs, depthwise separable only uses about 20% of the peak Flops compared to 94.5% for normal convolution. Nevertheless, the depthwise separable layer in Halide is still about 2 times faster than the popular deep learning framework TensorFlow. **We want to answer the following questions:**

1. Why is Halide slow for depthwise separable layers?
2. What are the aspects that prevent Halide from reaching a higher GPU utilization rate?
3. The implementation fuses depthwise convolution with point-wise convolution; does that make sense?
4. Can we do better?

Tools

Nvidia Nsight, Halide, Tensorflow, RTX 3070 x 1, RTX 2060 x 1, RTX 2080 Ti x 1.