README.md

Final Project

- Tuan Nguyen
- Spring 2017
- Advisor: Dr. Martin Hagan

Project Proposal/Abstract

Motivation

So far in this course, we worked mostly with single precision floating point (FP) and sometimes with double precision FP. However, some sources [9, 12, 13] claimed that low precision is sufficient for both training and running trained networks. Moreover, in 2016, the newest NVIDIA Pascal GPU architecture and CUDA 8 introduced half-precision floating point and even 8/16-bit integer computing capabilities. I myself followed this trend and realized that Torch also suported half-precision floating point three months ago. Caffe and TensorFlow also introduced half-precision support recently (but I do not know when). My research is about hardware design and adapting High Speed Arithmetics for different applications including Deep Learning. So I am excited to exploit and confirm how low precision floating point multipliers can be applied to Deep Learning. Because multipliers, especially floating point multipliers, consume the most space and power among arithmetic units, a saving in this operator can help save both memory and computing resources[11].

Problem Summary

In the scope and the timing budget of this project, I focused on examining the effect of low precision floating point multipliers (half-precision) in one Convolutional Neural Network on one dataset, comparing its performance with higher precision ones (single precision and double precision).

Dataset

I chose CIFAR-10 [4] dataset to train, validate and test the network. The reason is that it is a well-known and clean dataset with minimum effort on pre-processing tasks. The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. It is large enough to train a deep network.

Deep Network & Framework

For the sake of simplicity, I selected Convolutional Neural Network. In specific, I will simplify LeNet network to make it feasible in time. I knew Python/Lua, so I planned to use either Torch or Theano (but not Caffe). The reason is that I want to hack into the core of the layers in network to change the precision.

Performance Metrics

To benchmarking between half-precision, single precision and double precision, I will use 3 criterias:

- Error Rates
- Time

Estimated Schedule

- Design: 1 week
- Implementation: 1 week

- Test/Debug: 1.5 week
- Collect Data/Report: 1.5 week

References

1. MNIS
2. ImageNet & Tiny ImageNet
3. The Street View House Numbers (SVHN) Dataset
4. CIFAR-10
5. Training and investigating Residual Nets
6. Deep Residual Learning for Image Recognition
7. CS231n: Convolutional Neural Networks for Visual Recognition
8. TensorFlow Convolutional Neural Networks
9. Deep Learning Multipliers
10. Mixed-Precision Programming with CUDA 8
11. Performance Benefits of Half Precision Floats
12. Why are Eight Bits Enough for Deep Neural Networks?
13. Fixed Point Quantization of Deep Convolutional Networks