EECE571T Assignement 1 Report

Dingqing Yang

April 2019

Part1

I implement backpropogation for a single hidden layer MLP with relu activation. It is well known as universal approximator [Sonoda and Murata, 2017] that can learn arbitrary mapping from inputs to outputs. Therefore, I decided to generate random input output samples and use backpropogation to learn this mapping. The size of the random generated dataset is chosen to be close to some small real world datasets like Boston Housing dataset[1]. Since I use this MLP to do regression, the loss is measured as mean squared error (MSE) instead of classification accuracy. In each iteration, the backprop will evaluate gradient and the network parameters are updated using gradient descent. With correct implementation, the MSE between predicted outputs and true outputs are expected to decrease in each iteration with small enough step size. A reference backprop is implemented in pure numpy first for sanity checking. Each building blocks is implemented and tested in *module* directory. Finally, part1 code is included in *simpleBackprop.cu*

Part2

I profiled part1 code and find that matrix multiplication kernel take the most of the runtime. I implement tiling for matrix multiplication using shared memory for each square sub matrix with zero padding in *matrix_mul_shared.cu* in *module* directory. Part2 code included in *sharedBackprop.cu* is similar to *simpleBackprop.cu* but with matrix multiplication kernel using shared memory.

References

Sho Sonoda and Noboru Murata. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233–268, 2017.

[1] Details see here: `http://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html`