ECE 408 Final Project Milestone 1

Team 7: Peiming Wu, Xiuyi Qin, Evan Chang, Rio Martinez

Description

In this project, we will develop a convolutional neural network to recognize handwritten digits on CUDA GPU. The basic idea is to take a large number of the dataset from the Modified National Institute of Standard and Technology (MNIST), and then develop a system that can learn from those training datasets.

The neural network we are going to build consists of the input layer, hidden layer, and output layer. The neural network can learn their weights and biases using the gradient descent algorithm. Then we use backpropagation to compute the gradients of the cost function.

Performance Optimizations using CUDA

Texture Memory use

Since the input images are fixed, the first thing we came up with is probably to put input images into constant memory to speed up the reading process. However, due the relatively small size of the constant memory, our 4,000 input images may run out of constant memory. Hence, we may want to use texture memory, which is also a type of read-only memory that could be used for general computing like constant memory even though its primary purpose is for graphic applications.[1]

Unrolling for loops

In the sequential implementation of character recognition, multiple nested for-loops are needed to individually process each pixel of the input image. On a CPU this is undesirable, because each iteration is executed serially. By using a CUDA-enabled device to perform our image processing, we can effectively “unroll” these nested for-loops by assigning each iteration to a single CUDA thread. These CUDA threads can be processed in parallel, which dramatically reduces the time needed to process large datasets such as MNIST.

Reference:

[1] Texture Memory in CUDA: What is Texture Memory in CUDA programming. (n.d.). Retrieved from http://cuda-programming.blogspot.com/2013/02/texture-memory-in-cuda-what-is-texture.html

[2][1] Kirk, D., & Hwu, W.-mei W. (2017). *Programming massively parallel processors: a hands-on approach*. Amsterdam ; Boston ; Heidelberg: Elsevier.