Brian Rodriguez
U0853593

Final Project Writeup

During the course of the past couple of weeks, I have experimented with two types of final projects and only one option has posed to be a possibility. The first project that I experimented with was recreating the vDNN, also known as a virtualized Deep Neural Network. The second, now my final project; a case study on data parallelization.

The first half of exploring my final project was tinkering with the idea behind the vDNN and trying to recreate it. The vDNN allows for efficient training on a GPU when neural networks are on the order of 16 GBs or more. This means that the whole neural network cannot fit on a single GPU. The vDNN introduces the idea of prefetching layers and data when the layers are needed so that the memory of the GPU can be utilized by more useful, computationally intensive work.

The vDNN did not feel overbearing at first but I quickly began to realize that the work to implement such a model may take months. I wanted to use a deep learning framework such as pytorch or caffe2 but using a framework actually stopped me from being able to implement the vDNN. The frameworks are a wrapper and prevent the developer from tampering with the lower details such as fetching and freeing data at will.

The deep learning frameworks only allow a developer to build a model then the framework takes care of loading data and running the model through training iterations. I learned this after sifting through source code and finding that the only thing a developer can do is use a different type of memory allocator such as CUB or OS caching. I was surprised to find that these were the only optimizations a developer can utilize.

After discovering these findings, I also thought of implementing my own neural network from scratch; implementing convolution, relu, max pooling, etc. but I quickly figured out that even this task would take months to get working efficiently. I exhausted all of the options that I had, regarding the vDNN.

I met with Vinu and discussed different options for final projects. He suggested I do a case study on data parallelization using pytorch. I thought this may be a great idea since data parallelization is a huge part of deep learning and also pertains to the vDNN in some aspects.

The case study will be the affects of using multiple GPUs to parallelize the work of training a convolutional neural network. The case study will go into details about how spreading the work across multiple GPUs can actually hinder the training process. It will illustrate that more does not mean better when it comes to training a neural network. I hope to depict a fine line between efficient training and when communication across multiple GPUs actually slows the learning down.

If I am able to gather the computing resources on the CHPC machines to conduct the case study properly, then I will hopefully show the parallelization of up to 6 or 8 GPUs at once. I hope to provide clean visuals of utilizing different sizes of GPUs and the costs that come into play when tacking on more GPUs.

I plan on using a dataset that I have been using for my research with Rajeev, which has a sufficient amount of images. The dataset consists of four different types of blood cells with different types of diseases. The images range between varying sizes but I will scale them down to 224x224. I will also be implementing the VGG-19 convolutional neural network in pytorch to conduct the case study.

The hope is to show that spreading the work to thinly across multiple GPUs may not always speed-up the training process. I am not sure how many GPUs are required to show the slow down but I hope it does not take more than 6 to 8 GPUs. I am also unsure if the CHPC will have enough resources for me to accurately conduct my study.