CSE 261: Sparse Convolution Viz
Kiefer Selmon

Project data description:
Using a tool called condensa I can prune values from a DNN model and retrain to recover accuracy resulting in a sparsely populated set of weights achieving similar accuracy. The resulting sparse model contains significantly less (<1%) data, which presents an opportunity for a sparse implementation which only computes the result for nonzero elements of the model. Unfortunately, such an implementation fails to leverage modern devices, which have been tailored to efficiently execute large, dense, highly data parallel operations. The long term vision of my project is to analyze the pros and cons of sparse implementations of such operations, and identify opportunities to exploit locally dense clusters of data in the model.

Initial Viz Goal and target audience:
I will be visualizing sparse weights in a single convolution layer of a DNN. The visualization will help guide the sparse implementation of the convolution operation, and will eventually accompany the performance statistics, as well as architectural implications of the implementation on various hardware. The data will be a sparse 4-d matrix. Since I am interested in the data at the filter level, it will be represented as a 2d matrix of 3x3 filters. I will be visualizing local sparsity (at the filter level) as well as global sparsity. Possible other areas of interest may be visualizing the filters as vectors alongside density information to inform a cache prefetcher, or to visualize the density implications of altering the pruning value. The target audience will be the reading committee of my masters project, who will be professors with a background in computer architecture and an insight into performance critical, concurrent software.

Creating the workload:
Using the pytorch profiler I selected a layer of Alexnet to use as my case study. I selected the 4th convolution layer based on it having the largest memory footprint as well as the 2nd largest execution time. I chose to use unstructured pruning which prunes at an element level (rather than removing entire filters), the result is a layer with ~99.3% sparsity. The weights are contained in a 4d matrix with the shape (256, 384, 3, 3) which I will be visualizing. In order to implement the operation such that I can exploit the sparsity in the weights, the weights will be stored in compressed sparse row format and I will write a convolution implementation that uses this data structure.

Viz based implementations:
The visualizations should guide the long term goal of the project in two main ways. First is visualization for load balancing within a heterogeneous architecture. The visualization should provide insight into how nonzero elements are distributed throughout the matrix, and thus how work can be distributed among threads. Second, the visualization can provide insight for possible heterogenous execution. If it is observed that a single sub matrix in the weights is significantly denser than the rest of the matrix, that region may be offloaded to the GPU, while the sparse region is executed on the CPU.