MPhil Project Proposal

Chihang Wang

26 November 2015

1 Problem Description

In recent years, deep neural networks(DNN) have been setting state-of-art performance record on various AI applications including: object classification, object detection, image caption generation, etc. As the dataset is getting ever larger, the network is becoming deeper, and the number of parameters is exploding. The VGG model performing only a 1000-class classification task [1] already requires 138 million weight units.

Simultaneously, the most demanding field for Deep Neural Network application is the mobile and embedding devices. For example, it will be desirable for Siri to be available in offline mode; and a self-driving car relies on embedded devices to perform object detection in real time. On the other hand, mobile devices typically have a small working memory, iPhone-6 for instance has only a DRAM of size 1GB. The memory demand of DNN is prohibitive to such devices. Furthermore, the main-memory access is expensive in terms of both energy consumption and clock cycles. To run a 1-billion connection network with 20 frame per second frequency, the memory access alone takes 12.8W of energy [3], which is well-beyond the energy envelope of a typical mobile devices.

As the number of operations required is proportional to the number of weights, increasing weights means the forwarding time for each pass will become slower and slower, which makes real-time processing harder.

2 Aims and Objectives

The over-parameterized nature of DNN means there are rooms for network compression. There have been active researches on different approaches to compress the Network. The aim of this project is to study the various approaches, analyzing their respective advantages and disadvantages, explore the scope for combining the different methodologies, as well as possible extensions. As a by-product, an efficient package to perform DNN compression is to be produced.

There are two major focuses on Neural Network compression researches. One brings compression to an extreme and uses technique like Huffman encoding [2] to reduce the number of bits required to store each weight. Such techniques often require a decoding stage later, which may hamper the speed of running time. This means that the compression trades the computation speed for memory.

On the other hand, there have been compression techniques focusing on improving the speed. Vanhoucke discussed weight quantization to 8-bits and reported 2x performance gain [4]. Jaderburg explored the use of low-rank approximation to deep convolutional neural network and reported 2.5x speedup with no loss in accuracy [5].

This project will mainly focus on compression which improves the speed while not affecting the prediction performance.

References

[1] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014; arXiv:1409.1556.

[2] Song Han, Huizi Mao, William J. Dally A Deep Neural Network Compression Pipeline: Pruning, Quantization, Huffman Encoding, 2015; arXiv:1510.00149.

[3] Song Han, Jeff Pool, John Tran, William J.Dally Learning both Weights and Connections for Efficient Neural Networks 2015; arXiv: 1506.02626.

[4] Vincent Vanhoucke, Andrew Senior, Mark Z.Mao Improving the speed of neural network on CPUs 2011;

[5] Max Jaderburg, Andrea Vedaldi, Andrew Zisserman Speeding up Convolutional Neural Networks with Low Rank Expansions, 2014; arXiv: 1405.3866