Deep Neural Network Based on FPGA

Name: Shuo Zhang
Advisor: Dr. Tarek M. Taha

Introduction

Deep learning is a class of mathematical algorithms that is now heavily used for Big Data analytics. These algorithms are based on very large scale neural networks. One of the key challenges with deep neural network (DNN) is that it requires massive computing power. At present clusters of high performance graphics cards designed primarily for computing (known as GPGPUs) are used for these tasks. A key problem with clusters of GPGPUs, is that they consume large amounts of energy, thus making it difficult to scale existing massive computing systems to future Big Data volumes.

Parallel Cognitive Systems Laboratory of University of Dayton had already completed a novel computing system to efficiently implement deep learning algorithms based on application specific integrated circuits (ASIC), which provides high performance at reasonably low power consumption. However, these are extremely expensive to fabricate as they are essentially custom built processors. The Field Programmable Gate Array (FPGA) is a type of integrated circuit that can be reconfigured to implement a large range of arbitrary functions according to application requirements. FPGAs are much cheaper than ASIC and consume less power than CPU and GPU. The objective of this project is to develop DNN based on FPGA. I will optimize the whole design to make it more suitable for the training and inference. Several pattern recognition applications which use deep learning will be used to test and evaluate the design.

Methodology

Figure 1 shows the block diagram of a neuron within this network. The neuron performs two types of operations, (1) a dot product of the inputs and the weights and (2) evaluation of an activation function. The dot product operation can be seen in Eq. (1). The activation function of the neuron is shown in in Eq. (2). In a multi-layer neural network, a non-linear activation function is desired.

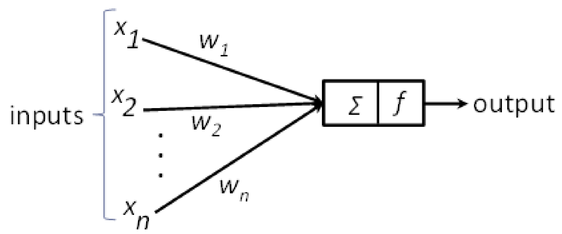

Figure 1. Neuron block diagram.

$$DP_j = \sum_{i=1}^{n} x_i W_{ij} \qquad (1)$$

$$y_j = f\left(DP_j\right) \qquad (2)$$

As shown in Fig. 2, this system consists of processing cores connected through an on-chip routing network, with one router per core.

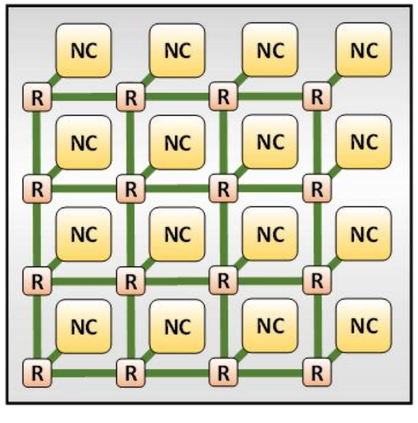

Figure 2. A multicore neuromorphic processor

Results

Figure 3 shows the throughput achieved on the RISC processor compared to the FPGA. The results show that the FPGA implementation provided about 16× and 300× higher throughput than the RISC processor for the edge detection and MINST applications respectively.

Application	RISC	FPGA	FPGA Efficiency
Edge detection (million pixels/sec)	0.393	6.56	16.7
MNIST (thousand digits/sec)	1.09	334	306.4

Figure 3. Application throughput.

Figure 4 compares the power consumed by the RISC processor and the FPGA. The results show that the RISC processor has an approximately 6× higher power consumption than the FPGA for both application.

Application	RISC (W)	FPGA (W)	FPGA Efficiency
Edge detection	0.9	0.145	6.2
MNIST	0.9	0.147	6.1

Figure 4. Application power consumption.

Conclusion

This project examined the design of a multicore neural processor on an FPGA. The implementation provided speedups of 16 and 300 times over a RISC core. Very high energy-delay efficiencies were seen for the FPGA. As future work, I will continue working on the training part of this deep neural network, and examine implementations on a larger FPGA to enable more processing cores.