Hardware/Software Co-verification of DNN for IoT Edge Devices: Leveraging FPGAs

By Katie Groves and Tolulope A. Odetola, Electrical and Computer Engineering Department

Abstract

Deep neural networks (DNN) is one of the emerging types of machine learning that is being used to solve problems that are too complex to be solved by humans. Field Programmable Gate Arrays (FPGAs) can be used to implement DNNs for IoTs because of their high throughput, low power operations, and portability. This research shows how one type of DNN can be placed on a PYNQ-Z1 board and maintain same prediction accuracy.

Introduction

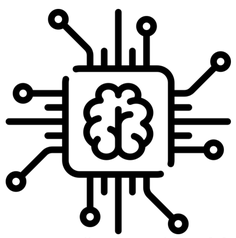

Fig. 1: Intelligent Hardware [1]

- There is a need for IoTs and edge devices to do real time classifications
- Currently, predictions and calculations for machine learning are sent through the network to perform computations off site and sent back to the device to output the answer.
- Applying a DNN directly to an IoT will eliminate the need for IoTs to be connected to a off site server and consequently save bandwidth along with reduced bandwidth

Need:

- With the need for internet and time taken to send and receive data, this method is too slow [2].
- Portability and flexibility of an FPGA with a DNN will allow for an adaption to changing environments and inputs from peripherals.
- When applying a deep learning architecture to an FPGA , a hardware/software co-verification is needed to ensure the accuracy is still maintained after implementation onto the hardware.

Research Question:

- How to accommodate for memory of storing weights and biases needed by the architecture and computationally intensive deep learning architectures into the FPGA?
- What methods to use to ensure a hardware/software co-verification technique?
- A solution to this problem would be to have an FPGA on the edge deceive to run computations and make predictions on site.
- A compression of the machine learning framework is needed in order to fit the architecture on the FPGA.

Method

Train	• Train the CNN on Caffe using a GPU
Test	• Test the weights to ensure correct prediction • Test reducing weight bit size and confirm its accuracy
C++	• C++ layer abstraction of the network • Convert parameters from python to C++
Implement	• Implement C++ layer abstraction in Vivado HLS • Ensure the code compiles with no errors
Test	• Read Test Image from File • Ensure it Mimics the Caffe output
Deploy	• Package Code and Generate custom IP in Vivado • Configure and deploy code to PYNQ-Z1

Results

For the following results, an image of a hand written 4, shown in Figure 2, was passed through both of the software and hardware networks. The results are shown below in Table 1.

Fig. 2: Hand written 4

	Caffe	Vivado
Conv1	[-3.50568414e-01, -3.50568414e-01, -3.50568414e-01, -3.50568414e-01, -3.55522346e+00, 5.44352293e+00, 4.20917273e-01, 7.09943485e+00, -4.79947901e+00, 1.08242369e+01, -2.91127090e+01, 3.29349670e+01, 1.54480698e+02, 3.58261383e+02, 2.97550690e+02, 1.78095383e+02, -2.38186407e+00, -7.85685241e-01, 2.56674469e-01, -1.86895883e+00, -1.06108618e+00, -1.33038926e+00, 1.98121774e+00, -4.44109738e-02]	-0.350568 -0.350568 -0.350568 -0.350568 -3.555223 5.443523 0.420917 7.099435 -4.799479 10.824235 -29.112707 32.934967 154.480698 358.261383 297.550659 178.095383 -2.381864 -0.785685 0.256675 -1.868959 -1.061086 -1.330389 1.981218 -0.044411
Pool1	[-3.50568414e-01, -3.50568414e-01, 5.44352293e+00, 7.09943485e+00, 1.08242369e+01, 1.26500420e+02, 4.74419556e+02, 3.92806396e+02, 7.68767166e+01, 4.71818209e+00, 8.91100407e+00, 3.66835737e+00]	-0.350568 -0.350568 5.443523 7.099435 10.824235 126.500420 474.419586 392.806366 76.876717 4.718182 8.911004 3.668357
Conv2	[3.61604843e+01, 1.76219711e+02, -1.28377762e+02, -4.49002869e+02, -3.12724609e+02, -1.60675400e+02, -1.87305954e+02, -1.28137436e+02], [-8.66113815e+01, -1.57161224e+02, -2.29579681e+02, -2.08867218e+02, -3.21133179e+02, -1.72957932e+02, -1.76641876e+02, -1.35855362e+02]]]], dtype=float32)	33.924271 173.983444 -130.613937 -451.239014 -314.960785 -162.911591 -189.542145 -130.373611 -88.847557 -159.397415 -231.815811 -211.103409 -323.369385 -175.194153 -138.091568 -138.091568
Predict	predicted class is #4.	prediction is 4

Table 1: Predicted results of both Caffe and Vivado

Discussion

- After taking the original CNN code and efficiently reducing the overall size of the code and the parameters, the code successfully ran in the program Vivado HLS.
- A 28x28 black and white image of a hand written number is called by the test bench code. The test bench code then passes the image to the CNN.
- A prediction with its predetermined weights is made. Given different hand written images, the code has successfully outputted the correct prediction for hand written numbers.

Conclusion/Future Works

It can be seen from our work that complex neural networks can be compressed onto hardware and still maintain accurate results just like its original software state.

Implementing the final CNN code on the PYNQ board help us with building a framework which will allow testing and verification of deep learning on a FPGA to be very convenient.

Future Work:

- Evaluating the effects of approximate computing on the accuracy of the deployed model on the PYNQ-Z1 board, shown in Figure 3.
- Coming up with a framework for hardware/software co-verification of DNN without requiring intricate knowledge of FPGA design.

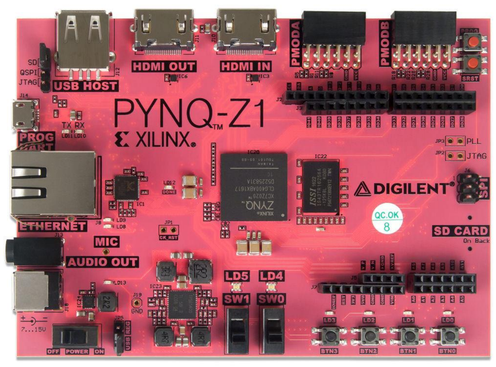

Fig. 3: Xilinx PYNQ-Z1 [3]

References

[1] Freepik. "Chip Free Vector Icons Designed by Freepik." *Flaticon*, www.flaticon.com/free-icon/chip_897219.

[2] H. Li, K. Ota, and M. Dong, "Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, Jan. 2018.

[3] "Python Zynq = PYNQ, Which Runs on Digilent's New $229 Pink PYNQ-Z1 Python Productivity Package." *Community Forums*, 28 June 2018, forums.xilinx.com/t5/Xcell-Daily-Blog-Archived/bg-p/Xcell/page/24.

[4] L. Stornaiuolo, M. Santambrogio, and D. Sciuto, "On How to Efficiently Implement Deep Learning Algorithms on PYNQ Platform," *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2018.

Acknowledgements: Dr. Hasan, the Electrical and Computer Engineering Department, and the Gower Fellowship
Contact: kmgroves42@students.tntech.edu