FPGA Accelerated Deep Learning Radio Modulation Classification Using MATLAB System Objects & PYNQ

Andrew Maclellan*, Lewis McLaughlin*, Louise Crockett, and Robert W. Stewart
Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, Scotland
Email: *{a.maclellan, lewis.mclaughlin}@strath.ac.uk

***Abstract*—Floating point Convolutional Neural Networks (CNNs) are computationally expensive and deeper networks can be impractical to deploy on FPGAs – consuming a large number of resources and power, as well as having lengthy development times. Previous work has shown that CNNs can be quantised heavily using fixed point arithmetic to combat this without significant loss in classification accuracy. We aim to quantise an existing CNN architecture for radio modulation classification to 2-bit weights and activations, while retaining a level of accuracy close to the original paper, for deployment on a Zynq System on Chip (SoC). To improve the development time for hardware synthesisable CNNs, we make use of MATLAB System Objects and HDL Coder. The PYNQ framework is presented as a practical means for accessing the functionality of the CNN. Our preliminary results show a high classification accuracy even with 2-bit weights and activations.**

I. Introduction and Motivation

Deep Learning (DL) and Artificial Intelligence (AI) have proven to be exciting and powerful machine learning-based techniques that have solved many real world challenges. Applications such as recognising and analysing images in computer vision and natural language processing are just a few examples of where DL has been highly effective.

With the increasing demand for efficient wireless data, high quality spectrum sensing, cognitive radio, and accurate channel estimation, smarter techniques are needed to handle the requirements of these areas. As a result DL has made appearances in the radio communications field where Deep Neural Networks (DNNs) and CNNs are trained with radio data rather than being expertly crafted, producing competing results to traditional techniques.

As DNNs and CNNs become more complex, their number of weights, layers, and computational cost increase, making them increasingly difficult to deploy for the high sample rate processing of radio applications. GPUs are typically used to train and deploy neural networks, but these provide a poor compute-to-power ratio compared to FPGAs. To support the high sample rates of radio, FPGAs have been extensively used to deploy signal processing algorithms in the past. In addition, FPGAs offer more flexibility with regard to data types, making them better suited to heavily quantised networks than GPUs.

Previous CNN FPGA implementations such as the FINN Framework by Umuroglu et al. [1], quantise a trained neural network down to 1-bit weights and activations and achieve throughput results of 12.3 million image classifications per second with 95.8% accuracy on an MNIST dataset. Quantising weights and activations down to 1-bit allows for efficient use of the FPGA fabric without sacrificing classification accuracy significantly. Another FPGA implementation of neural networks is the Ternary Neural Network by Alemdar et al. [2], which adopts a teacher and student technique of training. The teacher network is trained at full precision on a CPU/GPU system with only activations being quantised in the range of (-1, 0, 1). The weights and input data are then quantised to the same range for the student network which is then deployed on the FPGA. The accuracy rate achieved on MNIST was 98.14% at 255,102 images per second.

Our aim is to deploy an accelerated, heavily quantised CNN (2-bit weights and activations) for Automatic Modulation Classification (AMC) that integrates with the PYNQ framework to selectively demodulate different modulation schemes. We chose 2-bit weights and activations to fully utilise the range of values available for 2-bits unlike the Ternary Neural Network with its (-1, 0, 1) range.

The PYNQ framework makes use of the Python programming language to enable easier development of designs targeting a Zynq SoC. In combining this highly productive language with an FPGA, various components can be hardware accelerated for real-time deployment (neural network, modulation/demodulation, pulse shaping etc.) and controlled easily through a Jupyter Notebook, similar to calling functions from a software library [3]. Python is currently a prevalent environment for developing DL algorithms. In addition, a PYNQ image exists for RFSoC – a SoC created specifically for interfacing with RF signals – making it a suitable framework for consolidating DL and radio communications processing.

II. Preliminary Architecture of Quantised CNN

The final aim for the architecture is to construct a hardware efficient CNN that performs modulation classification in real time. As a step towards achieving this, a neural network was trained using QPSK and QAM-16 data and the weights and biases were saved to be quantised. A MATLAB HDL Coder System Object was constructed where dense and convolution layers can be loaded with weights and biases. These can then be quantised to 2-bits at synthesis time.

A. Training Neural Network

The CNN was constructed and trained using PyTorch. PyTorch is a deep learning framework that simplifies neural network training and deployment. The AMC CNN structure is based on a model proposed in [4]. The model consists of six layers in total, depicted in Table I, describing the activations, dimensions and total number of multiply-accumulates (MACs) for each layer.

TABLE I
NEURAL NETWORK PARAMETERS

Layer #	Layer type	Neurons	Activations	MACs
1	Input	$2 * 128$	-	-
2	Conv	$64 * 1 * 3$	ReLU	48384
3	Conv	$16 * 2 * 3$	ReLU	761856
4	Dense	128	ReLU	253952
5	Dense	2	Softmax	256
6	Output	2	-	-

The model was trained to classify 128 samples of modulated complex data and distinguish between QPSK and QAM-16. A set of 34,000, 128 long complex samples were used and simulated at varying SNRs. Once the model was trained, the weights and biases were saved. An SNR of 10dB was later used for testing the accuracy.

B. FPGA Architecture Design

A MATLAB System Object was used to convert the CNN architecture to HDL. The convolutional and dense layers were created within the System Object and programmed as a low-level design and in a hardware compatible coding style. A System Object design was used to enable layer weights to be loaded from PyTorch and be automatically quantised. Additionally, System Objects can generate HDL using HDL Coder within MATLAB. This allows for simulations of the quantised system before synthesis, saving development time.

Currently, we have created a System Object that takes in trained layer weights as parameters and quantises them to 2-bits. We believe this approach can result in greater accuracy while still maintaining low resource utilisation. The model was trained in PyTorch while limiting the weights during training to a 2-bit range and performing the back-propagation with the straight-through technique [5].

Once the preliminary design was created, accuracy tests were performed. The test compared the accuracy of the CNN at full precision with the 2-bit quantised version of the CNN. All input samples into the quantised CNN were set to a word length of 16 with 8 fractional bits. These results can be seen in Table II.

TABLE II
FULL PRECISION VS FIXED POINT CNN ACCURACY.

CNN Type	W and A prec.	Input prec.	Acc
Full precision CNN	float	float	99.2%
Fixed precision CNN	2-bit	16-bit	97.2%

The 2% accuracy drop shown in Table II indicates that the quantised weights do not significantly alter the accuracy of the system. This is the result of limiting the weight update during training to a range of 2-bits.

III. PROPOSED AND FUTURE WORK

Final Goal: To produce a fully configurable System Object that allows for loading of weights and automatic fixed point precision conversion that can be interfaced with PYNQ to provide a callable hardware-accelerated function. Figure 1 shows an example application where the CNN interfaces with PYNQ for demodulation analysis.

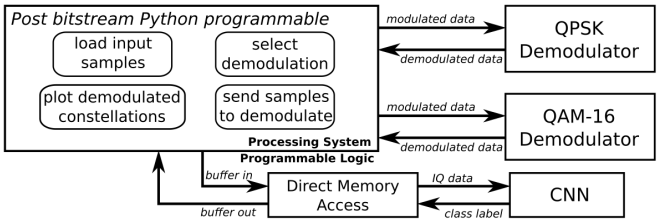

Fig. 1. Brief architecture of how CNN interfaces with Python within PYNQ.

We next aim to deploy and test the CNN on a Zynq 7020, using the Pynq-Z2 development board, and investigate various methods for reducing the resources required. Once an acceptable resource utilisation and timing has been achieved, we intend to scale up the project such that the neural network can classify over the full range of modulation schemes specified in [4]. This will be deployed on the RFSoC, leveraging the RF-ADCs to allow signals to be modulated, transmitted, received, classified and demodulated accordingly.

IV. CONCLUSION

Previous work shows that neural networks can be heavily quantised for implementation on FPGAs to greatly improve on resource utilisation and power efficiency without significantly lowering accuracy. We are motivated by the results shown in these papers and have began to investigate how this can be applied to radio communications. Preliminary results show high accuracy results even with reduced precision weights. We hope to demonstrate the suitability of MATLAB System Objects and the PYNQ framework for implementing and accessing the functionality of CNNs on FPGAs.

REFERENCES

[1] Y. Umuroglu, N. J. Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre, and K. Vissers, "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference," in *International Symposium on Field-Programmable Gate Arrays*, 2017.

[2] H. Alemdar, V. Leroy, A. Prost-Boucle, and F. Petrot, "Ternary neural networks for resource-efficient AI applications," in *International Joint Conference on Neural Networks*, 2017, pp.2547–2554.

[3] *PYNQ*, P. Lysaght, C. McCabe et al., Chapter 22 in *Exploring Zynq MPSoC: With PYNQ and Machine Learning Applications*, Strathclyde Academic Media, 2019.

[4] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *Communications in Computer and Information Science*, 2016.

[5] B. Moons, K. Goetschalckx, N. Van. Berckelaer, and M. Verhelst, "Minimum Energy Quantized Neural Networks," in *51st Asilomar Conference on Signals, Systems and Computers*, 2017.