Name of applicant: Yarib Israel Nevarez Esparza

Short title of proposal: Energy-efficient neural network processor for real-time deep learning applications.

1 Project idea

This project focuses on the research of design methodologies to improve the speed and energy efficiency of deep learning hardware processors. The motivation is to accelerate state-of-the-art deep neural networks in the edge to reduce the carbon footprint, the latency and the security-risks associated with the use of cloud-based AI solutions. In this research, hardware design methodologies based on approximation techniques are proposed to enhance processing efficiency of neural network algorithms. The research from this project is a fundamental part for future DFG proposals in the area of low-power design for neural network processors.

2 Summary

We have developed several concepts to use approximate techniques for neural networks [1], [2]. The purpose of this project is to implement and evaluate those concepts in practical state-of-the-art deep learning applications, specially those targeting edge-applications. As proof of concept, we select four practical applications to evaluate our implementation: (1) face mask detection [3],[4], (2) video surveillance [5], (3) advanced driver assistance system (ADAS) [6], [7], and (4) semantic segmentation for autonomous driving [8],[9]. These applications will be implemented by expanding our current research from embedded devices to edge devices with larger hardware resources. The results will be presented in conference and journal publications. This project will be available for the research community as an open source project; this includes hardware/software python interfaces, user documentation, and example designs to facilitate reusability in user-made neural network applications.

3 Description of proposal

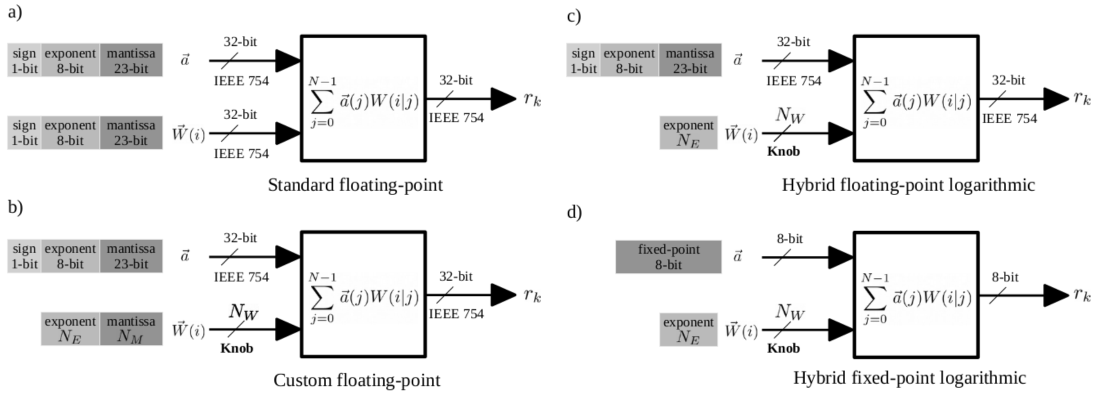
Deep convolutional neural networks (CNNs) are identified by their exceptional performance in computer vision in both research and industrial applications. For example, image-based disease detection in medical applications [10], inspection systems in agriculture [11], monitoring in smart industry [12], [13], and self-driving cars in automotive industry [14], [15]. However, the state-of-the-art of CNN-based algorithms, such as object detection models [16], [17], are characterized by their elevated memory and computational costs. Hence, the applicability of these algorithms is restricted to high performance computers equipped with power hungry processing units (e.g., GPUs and TPUs). This aspect represents the main constrain for efficient real-time performance of these algorithms in devices with limited resources [18], [19].

In order to enable real-time performance of state-of-the-art CNN-based algorithms in edge devices, we propose to use approximation techniques. Based on the intrinsic error-resilience of image processing and machine learning algorithms [20], [21], we propose approximate processing as the main optimization approach for this project.

Approximate computing has been used in a wide range of applications to optimize computational performance [22]. For neural network applications, two main optimization strategies are used, namely network compression and classical approximate computing [23]. The method known as *network compression* or *quantization* focuses on lowering the precision of weights and activation maps to shrink the memory footprint of the large number of parameters of neural networks [24], in addition to quantization, *network pruning* reduces the model size by removing structural portions of the parameters and its associated computations [25]. While on the other hand, the classical approximate computing consists of designing processing units that approximate their computation by employing simplified logic blocks [20], [22]. This technique accelerates computation and reduces energy consumption in error-resilient compute applications.

In previous research, we applied approximate processing to accelerate Spike-by-Spike (SbS) neural networks on embedded FPGA. We implemented a dedicated hardware module for vector dot-product computation, this hardware implements approximate computing with hybrid custom floating-point and logarithmic number representations. This design is configurable based on the bit truncation of the synaptic-weight vectors. **Fig. 1.** illustrates the vector dot-product hardware module with standard floating-point (IEEE 754) arithmetic and our approach with hybrid custom floating-point as well as logarithmic approximations. As a design parameter, the mantissa bit-width of the weight vector provides a tunable knob to trade-off between efficiency and quality of result (QoR). Since the lower-order bits have smaller significance than the higher-order bits, truncating them may have only a minor impact on QoR [20]. Further on, we can remove completely the mantissa bits in order to use only the exponent of a floating-point representation. Therefore, the most efficient setup becomes a logarithmic representation, which consequently leads to significant architectural hardware level optimizations using only adders and barrel shifters for vector dot-product approximation. Moreover, since approximations and noise have qualitatively the same effect [26], we apply noise tolerance plots as an intuitive visual measure to provide insights into the quality degradation of neural networks under approximate processing effects. In a previous research, our design accelerates SbS neural network computation by 20.5× and reduces the weight memory by 8×, with less than 0.5% of accuracy loss in the machine learning task.

Fig. 1. Dot-product hardware module with (a) standard floating-point (IEEE 754) arithmetic, (b) hybrid custom floating-point approximation, (c) hybrid floating-point logarithmic approximation, and (d) hybrid fixed-point logarithmic approximation.



3.1 Purpose of Impulse application

The purpose of this project is to evaluate the implementation of energy-efficient neural network processors for custom applications. As one of the objectives of my PhD project, I developed a fully functional and scalable hardware architecture for computing SbS networks in embedded systems [1]. This hardware architecture is optimized with a computational module with hybrid custom floating-point and logarithmic vector dot-product approximation. The vector dot-product is a computational block widely used in neural networks and in image/video processing algorithms [27], [28]. The purpose of this project is to implement and evaluate the performance of neural network processors with optimized hardware in practical state-of-the-art deep learning applications.

Aside from my doctoral project, the evaluation of our proven hardware design techniques on practical CNN applications represents a promising contribution to the field of hardware architectures for machine learning on edge devices. We selected four practical applications to evaluate our design, this will be carried out as doctoral research. The results will be reviewed and remarkable findings will be presented in conference and journal publications. This will contribute to my doctoral dissertation, to state-of-the-art knowledge, and to the research community.

3.2 Project implementation

For the project implementation, we initially design a neural network processor with our hardware optimization techniques (approximate computing). This hardware design is implemented in the edge FPGA. To use the neural network processor on the edge FPGA, we implement a python software infrastructure to facilitate user interface and usability. Afterwards, we evaluate the performance of our hardware implementation with focus on speed and power dissipation. Finally, we present remarkable findings in conference and journal publications, and we release the project for the community as an open source project. The start date is on 15/Jun/2021 and the end date is on 27/May/2022.

4 Cooperations

There is no third party cooperation.

5 Links to other projects receiving third-party funding

My Ph.D. is sponsored by the Consejo Nacional de Ciencia y Tecnologia – CONACYT (the Mexican National Council for Science and Technology).

6 Costs

Additional experiments incur in the following costs for specialized-equipment.

6.1 Outline of costs

Item	Quantity	Description	Unit price	Amount
1	1	Zynq UltraScale+ MPSoC ZCU104 Evaluation Kit https://www.xilinx.com/products/boards-and-kits/zcu104.html	€1,062.63	€1,062.63
2	1	Ultra96-V2 Zynq UltraScale+ ZU3EG Dev. board. https://de.farnell.com/avnet/aes-ultra96-v2-g/sbc-arm-cortex-a53-cortex-r5/dp/3050481	€202.67	€202.67
3	1	USB to JTAG/UART adapter for Ultra96 Dev. board. https://de.farnell.com/yageo/aes-acc-u96-jtag/usb-zu-jtag-uart-pod/dp/2915522?MER=sy-me-pd-mi-acce	€36.06	€36.06
4	1	Power supply kit, 12 V, 4 A, for Ultra96 Dev. boards. https://de.farnell.com/votoo/vp-1204000/netzteil-kit-12v-4a/dp/2921438?MER=sy-me-pd-mi-acce	€19.95	€19.95
5	2	Webcam, HD Pro, 1280 x 720p resolution, 3MP. https://de.farnell.com/en-DE/logitech/960-001063/hd-pro-webcam-3mp-720p/dp/2675982?st=webcam	€34.79	€69.58
			Total	**€1,390.89**

7 References

[1] Nevarez, Yarib, et al. "Accelerator Framework of Spike-By-Spike Neural Networks for Inference and Incremental Learning in Embedded Systems." *2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. IEEE, 2020.

[2] Najafi, Ardalan, et al. "Coherent design of hybrid approximate adders: Unified design framework and metrics." *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 8.4 (2018): 736-745.

[3] Sharma, Vinay. "Face Mask Detection using YOLOv5 for COVID-19." (2020).

[4] Loey, Mohamed, et al. "Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection." *Sustainable Cities and Society* (2020): 102600.

[5] Molchanov, V. V., et al. "Pedestrian detection in video surveillance using fully convolutional yolo neural network." *Automated visual inspection and machine vision II*. Vol. 10334. International Society for Optics and Photonics, 2017.

[6] Putra, M. H., et al. "Convolutional neural network for person and car detection using yolo framework." *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)* 10.1-7 (2018): 67-71.

[7] Lai, Chen-Wei, et al. "Vision based ADAS for Forward Vehicle Detection using Convolutional Neural Networks and Motion Tracking." *VEHITS*. 2019.

[8] Simon, Martin, et al. "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019.

[9] Feng, Di, et al. "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges." *IEEE Transactions on Intelligent Transportation Systems* (2020).

[10] Islam, Md Zabirul, Md Milon Islam, and Amanullah Asraf. "A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images." *Informatics in medicine unlocked* 20 (2020): 100412.

[11] Wan, Shaohua, and Sotirios Goudos. "Faster R-CNN for multi-class fruit detection using a robotic vision system." *Computer Networks* 168 (2020): 107036.

[12] Li, Liangzhi, Kaoru Ota, and Mianxiong Dong. "Deep learning for smart industry: Efficient manufacture inspection system with fog computing." *IEEE Transactions on Industrial Informatics* 14.10 (2018): 4665-4673.

[13] Lee, Ki Bum, Sejune Cheon, and Chang Ouk Kim. "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes." *IEEE Transactions on Semiconductor Manufacturing* 30.2 (2017): 135-142.

[14] Ouyang, Zhenchao, et al. "Deep CNN-based real-time traffic light detector for self-driving vehicles." *IEEE transactions on Mobile Computing* 19.2 (2019): 300-313.

[15] Valiente, Rodolfo, et al. "Controlling steering angle for cooperative self-driving vehicles utilizing cnn and lstm-based deep networks." *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019.

[16] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection." *arXiv preprint arXiv:2004.10934* (2020).

[17] Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.

[18] Al-Turjman, Fadi, ed. *Artificial intelligence in IoT*. Springer, 2019.

[19] Ahmad, Ijaz, et al. "Challenges of AI in wireless networks for IoT." *arXiv preprint arXiv:2007.04705* (2020).

[20] Hanif, Muhammad Abdullah, Rehan Hafiz, and Muhammad Shafique. "Error resilience analysis for systematically employing approximate computing in convolutional neural networks." *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018.

[21] Du, Zidong, et al. "Leveraging the error resilience of machine-learning applications for designing highly energy efficient accelerators." *2014 19th Asia and South Pacific design automation conference (ASP-DAC)*. IEEE, 2014.

[22] Han, Jie, and Michael Orshansky. "Approximate computing: An emerging paradigm for energy-efficient design." *2013 18th IEEE European Test Symposium (ETS)*. IEEE, 2013.

[23] Bouvier, Maxence, et al. "Spiking neural networks hardware implementations and challenges: A survey." *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 15.2 (2019): 1-35.

[24]Zhou, Aojun, et al. "Incremental network quantization: Towards lossless cnns with low-precision weights." *arXiv preprint arXiv:1702.03044* (2017).

[25] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." *arXiv preprint arXiv:1510.00149* (2015).

[26] Venkataramani, Swagath, et al. "Approximate computing and the quest for computing efficiency." *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2015.

[27] Hu, Miao, et al. "Dot-product engine as computing memory to accelerate machine learning algorithms." *2016 17th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2016.

[28] Wilson, Joseph N., and Gerhard X. Ritter. *Handbook of computer vision algorithms in image algebra*. CRC press, 2000.