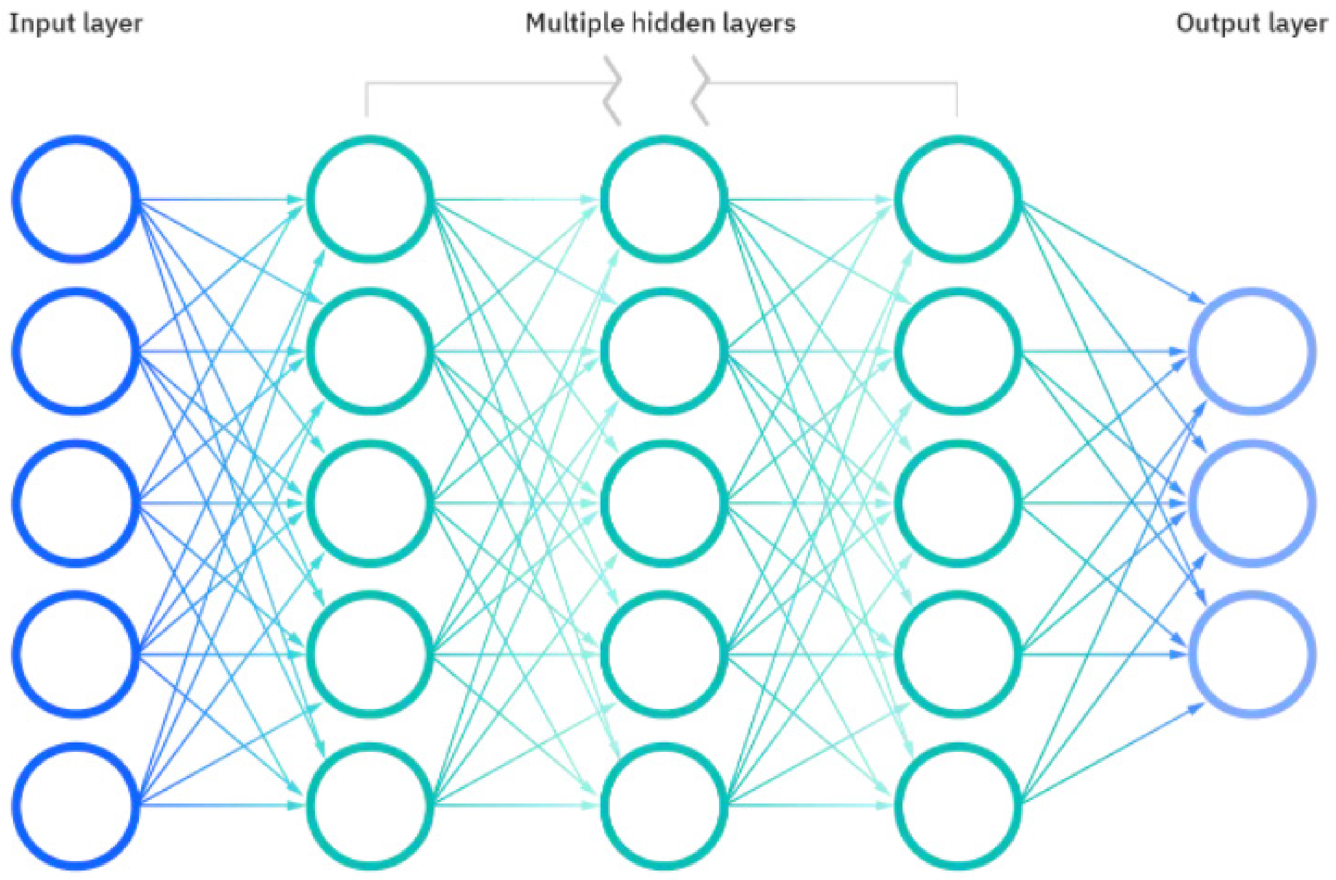
VIVIDSPARKS

Deep Neural Network Using FalKon



Introduction

Neural Networks (NN) are highly parallel workloads which require massive computational resources for training and often utilize customized accelerators such as Google's Tensor Processing Unit (TPU) to improve the latency, or reconfigurable devices like FPGAs to mitigate power bottlenecks, or targeted ASICs such as Intel's Nervana to optimize the overall performance. For deploying NN algorithms on the end-device (e.g. AI on the edge, IoT), memory and performance become prohibitive.

Convolutional Neural Network (CNN) is a special class of multi-layer neural networks, designed to recognize and analyze visual patterns directly from pixel images. They are usually comprised of a sequence of convolutional layers, activation functions, pooling layers, fully connected layers and normalization layers. In this study, we focused on five CNNs: (1) Inception-v2: a 22 layers network that introduces a special 1x1 convolution, and using global average pooling instead of using fully connected layers [1]. (2) ResNet-50, short for Residual Network. It introduces the idea of (identity shortcut connection) that skips one or more layers to address the vanishing gradient problem [2]. It is a deep residual network of 50 layers. (3) ResNet-18: one of the residual network variants. It consists of 18 layers. (4) MobileNet: is a small model built upon the idea of using depthwise separable convolutions as ecient building blocks [3]. (5) SqueezeNet: is a family of models that achieve AlexNet-level accuracy on ImageNet with 50x fewer parameters [4]. In this work, we evaluate the performance of neural network inference implementation of [Inception-v2 and ResNet-50, ResNet-18, Mobilenet-v2 and SqueezeNet].

Methodology

Accuracy: classification accuracy (test error rate) is used to evaluate the GPU and FPGA implementations by comparing their results to the ground truth in the whole Imagenet validation set with 50k images.

System performance: which measures the performance of the complete pipeline, including initialization, image reading, pre-processing, post-processing and data transfer. It measures the un-optimized performance of the platform by capturing data movement bottlenecks. Figure 1 shows steps involved in each of the compute and systems performance measurement.

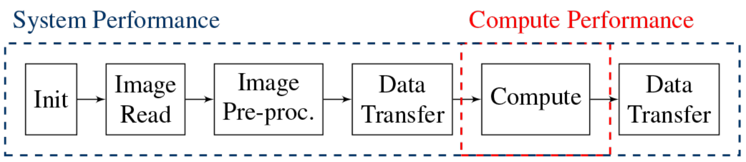


Figure 1: Steps involved in compute and system performance measurements

Energy: Energy consumption per frame quantifies the amount of electrical energy dissipated by hardware accelerators to perform a kernel's operations on one frame. It is measured as the power consumed during the delay time to process a frame. Device power can be divided in two parts: (1) Static power: represents the amount of power consumed when no active computation is taking place (system is idle), (2) Dynamic power: represents the amount of power consumed above the static power level when the system is computing.

Energy-Delay Product (EDP): Energy per frame alone do not show the entire picture. A hardware platform can be extremely low power while being too slow to be of practical use. The Energy Delay Product (EDP) [5] metric takes into account the throughput of the algorithm measured in (ms/frame) along with the energy consumed per frame (mJ/frame). EDP is the product of energy/frame and delay time. This way, a fair comparison can be made when deciding which hardware architecture is better suited for specific computation. Lower EDP is better which means that the hardware architecture can finish specific computation tasks using less power in less time.

Measurement Techniques and Platforms: In this work, we evaluated two popular platforms for deploying embedded vision applications: NVIDIA RTX 3090 and Xilinx Alveo U200 card. These platforms come equipped with an on-board power measuring IC that can measure multiple power rails such as: CPU cores and GPU cores on the Jetson, and programmable logic, full power CPU cores and low power CPU cores on the FPGA platform.

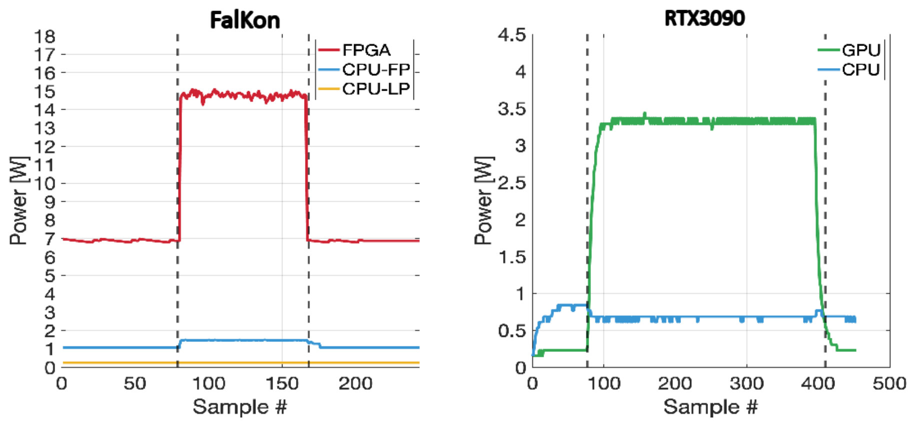


Figure 2: Measuring power samples on the platform's CPU cores (first 1000 frames), and its FalKon or GPU (second 1000 frames).

For every benchmark, we first processed 1000 frames on the CPU core of the platform and then 1000 frames on the hardware accelerated part of the platform. This can be seen in Figure 2, where the first two vertical lines mark the first 1000 frames on the CPU and the following two lines mark the last 1000 frames on the hardware. We computed the average frame rate by measuring the time between vertical lines and divided it by 1000. The x-axis represents the number of power samples taken for each platform. Note that the Alveo FPGA card has a different sampling rate than the RTX 3090. Frames were in RGB with 224224 resolution. We also used 1024 frames instead 1000 frames to have multiple of batch sizes.

Hardware Environments: In this work, we used Xilinx Alveo U200 FPGA board. It has a 16nm FPGA chip an on-board 32GB DDR4 RAM with a peak bandwidth of 77Gb/s with clock frequency of 300MHz. For GPU board, we used the NVIDIA GEFORCE RTX3090 (10496 cores) has 24GB DDR6X RAM with clock frequency of 1.70GHz.

Software Environments: On both platforms, CNN are coded in PyTorch.

System Performance: System performance measures the efficiency of the complete inference pipeline including its pre-processing, computation, data movement, and post-processing stages which gives insight into the actual performance achieved after deployment. It captures potential memory bandwidth bottlenecks in data copying between o-chip and on-chip memories.

In this work, we measure the performance of GPU and FalKon FPGA implementations of Inception-v2, ResNet-50, ResNet-18, MobileNet-v2 and SqueezeNet networks. We measure their performance at multiple batch sizes (b=1,b=2, ..., b=128) and thread counts (t=1, t=2, ..., t=8) to evaluate the eect of increasing batch size on data reuse and data movements reduction. Figure 3 and 4 show the experimental results of Inception-v2 and ResNet-50. The blue, green and red bars represents, GPU and FalKon, respectively. We measured the frame rate and EDP at different batch sizes and thread counts. Then, we selected the highest frame rate and lowest EDP among them for the side-to-side comparison.

The results show that, for Inception-v2, the FalKon (t=6) achieves a speed up of 2.5X to the GPU (b=128) For ResNet50, the FalKon (t=8) also achieves a speed up of 2.1X, to the GPU FP16 (b=128), respectively. This speed-up comes from the low numerical precision (Posit 8) used in FalKon compared to FP16 in GPU, Another observations is that the FalKon implementations of Inception-v2 and ResNet50 are 4.76X and 3.9X more energy efficient when number of threads equals (t=8) compared to (t=1).

In terms of EDP, the FalKon implementations have lower EDP values compared to the CPU and GPU FP16 implementations. Figure 3 shows that FPGA implementation (t=8) of Inceptionv2 has an EDP reduction ratios of 1.5X compared to the GPU FP16 (b=128). Figure 4 shows that FalKon implementation (t=8) of ResNet50 has 1.1X EDP reduction ratios compared to the GPU FP16 (b=128).

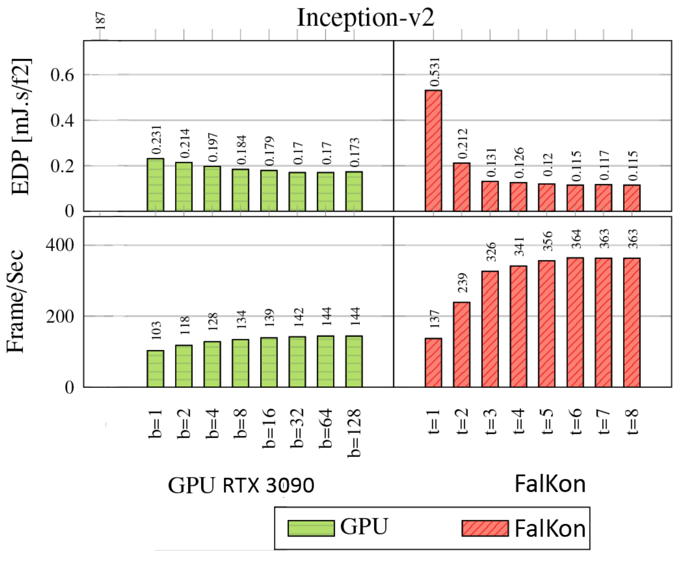


Figure 3: [System Performance] A comparison between GPU and FalKon in terms of frame Rate (fps) and energy delay product (EDP) for Inception-v2 Network.

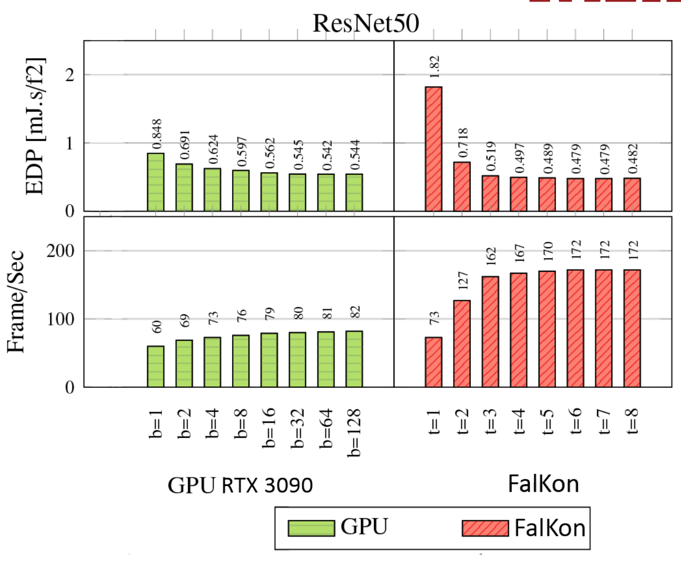


Figure 4: [System Performance] A comparison between GPU and FalKon in terms of frame Rate (fps) and energy delay product (EDP) for ResNet50 Network.

Figure 5,7 and 6 show the experimental results for the small networks: ResNet-18, Mobilenetv2 and SqueezeNet. For ResNet-18, the FalKon (t=6) achieves a speed up of 2.6X compared to the GPU FP16 (b=128). For Mobilenetv2, the FalKon (t=7) also achieves a speed up of 2.9X, compared to the GPU FP16 (b=128). For SqueezeNet, the FalKon (t=6) achieves a speed up of 2.5X compared to the GPU FP16 (b=128).

Table 1 summarizes the FalKon's frame rate and EDP reduction ratio compared to the GPU FP16 implementations. The FalKon is 2.1X–2.9X faster and 1.1X–2.2X more energy efficient than GPU FP16 implementations when running Inception-v2, ResNet-50, ResNet-18, Mobilenetv2 and SqueezeNet.

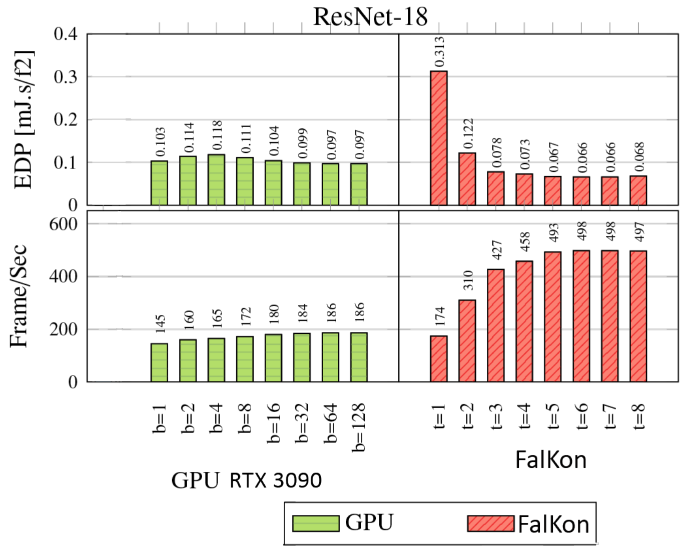


Figure 5: [System Performance] A comparison between GPU and FalKon in terms of frame Rate (fps) and energy delay product (EDP) for ResNet-18 Network.

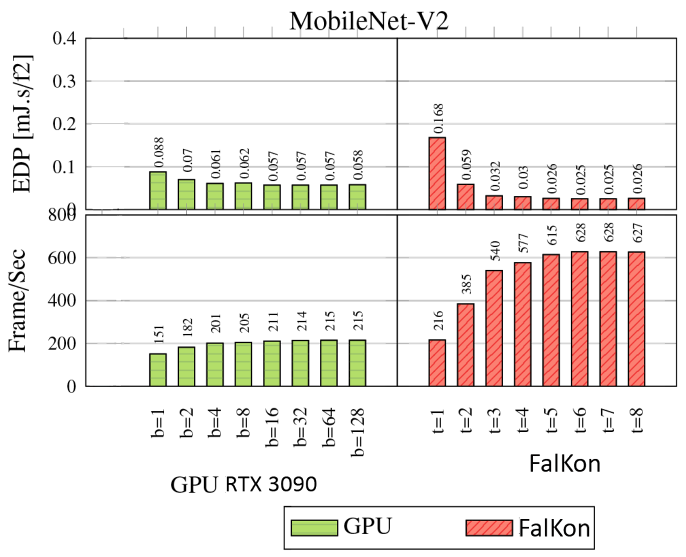


Figure 7: [System Performance] A comparison between GPU and FalKon in terms of frame Rate (fps) and energy delay product (EDP) for MobileNet Network.

Conclusion

In this work we performed in-depth benchmark analysis of two embedded platforms, GPU- and FPGA-accelerated, evaluating the efficiency of their different hardware architectures towards neural networks [Inception-v2, ResNet-50, ResNet-18, MobileNet-v2 and SqueezeNet]. Given the energy-efficiency focus, three key metrics are collected in the benchmarks: energy per frame, frame rate and energy delay product (EDP). The FalKon is 2.1X–2.9X faster and 1.1X–2.2X more energy efficient than GPU implementations when running Inception-v2, ResNet-50, ResNet-18, Mobilenetv2 and SqueezeNet.

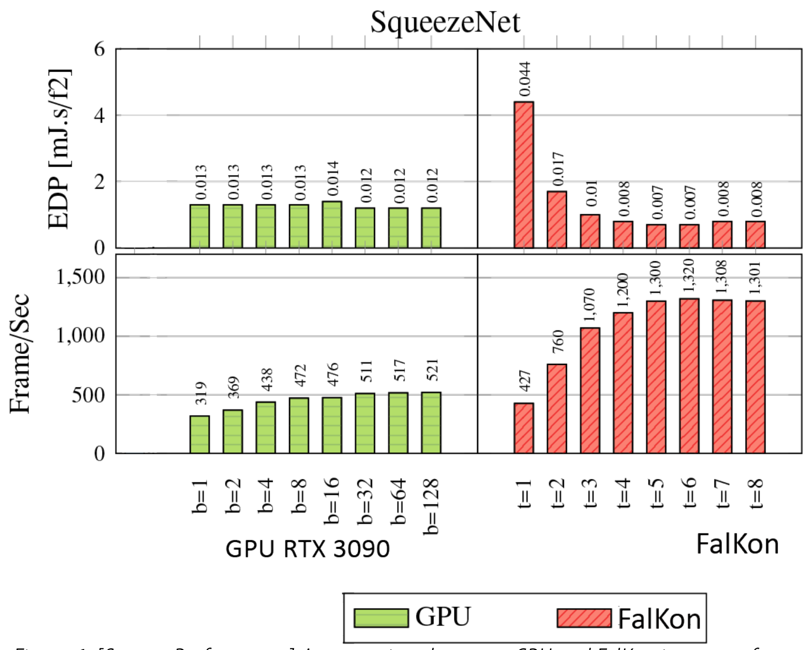


Figure 6: [System Performance] A comparison between GPU and FalKon in terms of frame Rate (fps) and energy delay product (EDP) for SqueezeNetV2 Network.

Model	Frame Rate (fps)	EDP (mJ.s/f2)
Inception-v2	2.5X	1.5X
ResNet-50	2.1X	1.1X
ResNet-18	2.6X	1.4X
Mobilenet-v2	2.9X	2.2X
SqueezeNet	2.5X	1.5X

Table 1: FalKon Speedup and EDP Reduction Ratios with Respect to GPU FP16 when [b=128 and t=8]

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[2] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4510– 4520.

[4] F. N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf,W. J. Dally, K. Keutzer, Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size, arXiv preprint arXiv:1602.07360 (2016).

[5] M. Horowitz, E. Alon, D. Patil, S. Naziger, R. Kumar, K. Bernstein, Scaling, power, and the future of cmos, in: Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International, IEEE, 2005.