Table 4: Inception V2 resource consumption

	BRAM	DSP	FF	LUT
Our work	3067	2041	539422	938159
	71%	32%	23%	79%

Table 5: Inception V2 performance comparison with previous implementations

	latency per image	latency speedup	power	energy efficiency
GPU	89.0 ms	2.16X	109.1 W	9.780 J/pic/s
FPT16 [19]	192.3 ms	1X (baseline)	N/A	N/A
FPGA17 [13]	83.6 ms	2.30X	13.2 W	1.175 J/pic/s
Our work	23.7 ms	8.11X	10.6 W	0.251 J/pic/s

Our implementation use 16-bit fixed point for both weight and feature map, and the operating frequency is 200 MHz. The resource consumption and simulation performance results are shown in Table 4 and Table 5.

We implement our design with Vivado HLS 2017.1 and find that our implementation tends to use more LUTs. This situation is due to the following reasons. First, both FFT and Winograd transformation comsume LUTs because multiplications are reduced to either additions or constant multiplications, which is implemented using LUTs. Second, the control logic in the Inception engine IP is more complicated compared to conventional convolution implementation, thus taking up more LUTs.

6.3 Evaluation

We first compare our implementation on FPGA with GPU result. We use a cutting-edge Pascal-based NVidia GTX 1080 GPU, which has a 8.9 TFLOPS peak performance. The GPU implmentation is on Torch, with CUDA 8.0. We also compare our work with the results reported in Zhang's work (FPGA2017) [13], and DiCecco's work (FPT2016) [19] which are works that evaluated GoogLeNet (Inception V1). These two works also implement fast convolution algorithms. FPGA2017 implements Overlap-Add FFT convolver on a CPU + FPGA system, and FPT2016 implements Winograd convolution on a Xilinx Virtex7 board. Our implementation is in fact Inception V2, which is the original Inception added with batch normalization layer after each CONV layer, thus has slightly more computations. The result is shown in Table 5. We use inference latency as the evaluation metric, since facial recognition/verification is a latency critical task. For works that don't report latency, we calculate single image latency by dividing the entire network computation operations with the reported GOPS. Our result shows that, compared with GPU, we achieve 3.75x latency improvement. For FPGA works, we achieve superior results, with 3.53x and 8.11x latency speed up compared to the FPGA2017 and FPT2016, respectively. We also achieve 4.68x better energy efficiency compared to FPGA2017.

7 CONCLUSIONS

In this paper, we explore different fast convolution algorithms including Winograd's minimum filter algorithm and FFT-based algorithm, and find the best strategy to apply them on different types of convolutions. We implement a configurable IP-based end-to-end CNN accelerator targeting FaceNet (Inception V2) using C-based HLS. Our solution surpasses both NVIDIA GTX 1080 GPU and previous FPGA results. We envision that such face recognition system can be paired with multiple low-power video capture systems, with the FPGA deployed in a central server and close to database, for fast real-time multi-face recognition and verification, to satisfy the need for security, border control, and other related applications.

ACKNOWLEDGMENTS

This work is supported by IBM-Illinois Center for Cognitive Computing Systems Research (C^3SR), a research collaboration as part of the IBM AI Horizons Network. We also thank Kyle Rupnow of Inspirit IoT Inc. for helpful discussions.

REFERENCES

[1] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.
[2] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.
[3] Yuhong Li, Xiaofan Zhang, and Deming Chen. CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. *CVPR*, 2018.
[4] Su Liu, Alexandros Papakonstantinou, Hongjun Wang, and Deming Chen. Real-time object tracking system on FPGAs. In *SAAHPC 2011*.
[5] Kyle Rupnow, Yun Liang, Yinan Li, Dongbo Min, Minh Do, and Deming Chen. High level synthesis of stereo matching: productivity, performance, and software constraints. In *FPT 2011*.
[6] Chun He, Alexandros Papakonstantinou, and Deming Chen. A novel SoC architecture on FPGA for ultra fast face detection. In *ICCD 2009*.
[7] Chen Zhang, Peng Li, Guangyu Sun, Yijin Guan, Bingjun Xiao, and Jason Cong. Optimizing FPGA-based accelerator design for deep convolutional neural networks. In *FPGA*, 2015.
[8] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
[9] Yongming Shen, Michael Ferdman, and Peter Milder. Overcoming resource underutilization in spatial cnn accelerators. In *FPGA*, 2016.
[10] Xiaofan Zhang, Xinheng Liu, Anand Ramachandran, Chuanhao Zhuge, Shibin Tang, Peng Ouyang, Zuofu Cheng, Kyle Rupnow, and Deming Chen. High-performance video content recognition with long-term recurrent convolutional network for FPGA. In *FPL*, 2017.
[11] Nicolas Vasilache, Jeff Johnson, Michael Mathieu, Soumith Chintala, Serkan Piantino, and Yann LeCun. Fast convolutional nets with fbfft: A GPU performance evaluation. *arXiv preprint arXiv:1412.7580*, 2014.
[12] Andrew Lavin and Scott Gray. Fast algorithms for convolutional neural networks. In *CVPR*, 2016.
[13] Chi Zhang and Viktor K Prasanna. Frequency domain acceleration of convolutional neural networks on CPU-FPGA shared memory system. In *FPGA*, 2017.
[14] Utku Aydonat, Shane O'Connell, Davor Capalija, Andrew C Ling, and Gordon R Chiu. An OpenCL (tm) deep learning accelerator on arria 10. *arXiv preprint arXiv:1701.03534*, 2017.
[15] Shmuel Winograd. Arithmetic complexity of computations, cbms-nsf regional conference series in applied mathematics. *SIAM*, 1980.
[16] Philipp Gysel, Mohammad Motamedi, and Soheil Ghiasi. Hardware-oriented approximation of convolutional neural networks. *arXiv:1604.03168*, 2016.
[17] Song Han, 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.
[18] Xiaofan Zhang, Anand Ramachandran, Chuanhao Zhuge, Di He, Wei Zuo, Zuofu Cheng, Kyle Rupnow, and Deming Chen. Machine learning on FPGAs to face the IoT revolution. In *ICCAD, 2017*. IEEE, 2017.
[19] Roberto DiCecco, Griffin Lacey, Jasmina Vasiljevic, Paul Chow, Graham Taylor, and Shawki Areibi. Caffeinated FPGAs: FPGA framework for convolutional neural networks. In *FPT 2016*. IEEE.