LightBulb: A Photonic-Nonvolatile-Memory-based Accelerator for Binarized Convolutional Neural Networks

Farzaneh Zokaee§ Qian Lou§ Nathan Youngblood‡ Weichen Liu♯ Yiyuan Xie† Lei Jiang§
§Indiana University Bloomington, USA ‡University of Pittsburgh, USA
♯Nanyang Technological University, Singapore †Southwest University, China
§{fzokaee, louqian, jiang60}@iu.edu ‡nathan.youngblood@pitt.edu ♯liu@ntu.edu.sg †yyxie@swu.edu.edu

***Abstract*—Although Convolutional Neural Networks (CNNs) have demonstrated the state-of-the-art inference accuracy in various intelligent applications, each CNN inference involves millions of expensive floating point multiply-accumulate (MAC) operations. To energy-efficiently process CNN inferences, prior work proposes an electro-optical accelerator to process power-of-2 quantized CNNs by electro-optical ripple-carry adders and optical binary shifters. The electro-optical accelerator also uses SRAM registers to store intermediate data. However, electro-optical ripple-carry adders and SRAMs seriously limit the operating frequency and inference throughput of the electro-optical accelerator, due to the long critical path of the adder and the long access latency of SRAMs. In this paper, we propose a photonic nonvolatile memory (NVM)-based accelerator, LightBulb, to process binarized CNNs by high frequency photonic XNOR gates and popcount units. LightBulb also adopts photonic racetrack memory to serve as input/output registers to achieve high operating frequency. Compared to prior electro-optical accelerators, on average, LightBulb improves the CNN inference throughput by $17\times \sim 173\times$ and the inference throughput per Watt by $17.5\times \sim 660\times$.**

***Index Terms*—Optical Accelerator, Photonic Racetrack Memory, Photonic Phase Change Memory, Binarized Neural Network**

I. Introduction

Due to their high accuracy, convolutional neural networks (CNNs) have been employed by cloud-based intelligent personal assistants, e.g., Apple Siri, Google Now, and Microsoft Cortana, to process a wide range of problems, e.g., object recognition, speech processing and machine translation. However, an inference of a CNN involves a large number of computationally expensive convolutions. For instance, an inference of AlexNet requires 724M floating point multiply-accumulate (MAC) operations. The essential computing effort of CNN inferences significantly increases the power consumption and carbon footprint of data centers.

Billions of inferences will be performed on a trained CNN model in data centers [8]. Instead of power-hungry GPUs, ASIC [7], FPGA [31], and ReRAM [25]-based accelerators are built to more energy-efficiently process CNN inferences. Recent works [17], [26], [16] present optical CNN accelerators to further improve the inference throughput. There are two types of optical CNN accelerators, i.e., the *all-optical* design [26], [16] and the *electro-optical* design [17]. Although optical CNN accelerators demonstrate up to 13× [17] inference throughput per Watt over various prior accelerator designs, due to the low power consumption of photonic devices, two types of optical CNN accelerators are constrained by different bottlenecks.

All-optical accelerators [26], [16] are limited by their low inference accuracy. They process an entire CNN all by photonic devices without involving SRAM registers, their inference accuracy is hurt by imperfections of photonic devices. Since all-optical accelerators use an analog optical signal to represent $8 \sim 16$ bits, the variations of photonic devices inevitably introduce and accumulate tiny errors in these analog optical signals, resulting in significant accuracy degradation. Even on small MNIST dataset, an all-optical accelerator obtains only 91.75% [16] inference accuracy, which is even lower than that achieved by a support vector machine [12].

The inference throughput of the electro-optical CNN accelerator [17] is limited by the long critical path of the electro-optical adders and the long SRAM access latency. Prior work [17] first quantizes the weights of a CNN into power-of-2 representations, so that expensive floating point MAC operations can be replaced by cheap fixed point additions and binary shifts. An electro-optical CNN accelerator [17] is proposed to process intensive fixed point additions and binary shifts by electro-optical ripple-carry adders and photonic binary shifters. The intermediate results are stored in SRAM-based input/output registers. Since an optical signal represents only 1 bit, the errors caused by optical device variations are trivial in the electro-optical accelerator. However, the long critical path of the electro-optical ripple-carry adders and the long SRAM access latency severely limit the operating frequency of the electro-optical accelerator, thereby significantly decreasing the CNN inference throughput per Watt.

In this paper, we propose an electro-optical CNN accelerator, *LightBulb*, to process the inferences of binarized CNNs by photonic XNOR and popcount units. LightBulb first binarizes the weights and activations of a CNN into linear combinations of {-1, +1}s, so the floating point MAC operations can be replaced by XNORs and popcounts. We propose a photonic microdisk-based XNOR gate and a photonic phase change memory (PCM)-based analog-to-digital converter (ADC) for LightBulb to accelerate XNORs and popcounts, respectively. Both our photonic XNOR gate and ADC can be operated at 50GHz. We further integrate photonic racetrack memory into LightBulb to serve as input and output registers to support the entire accelerator to work at such high frequency. Our

This work was supported in part by NSF CCF-1908992 and CCF-1909509. Weichen Liu was supported by NAP M4082282 and SUG M4082087. Yiyuan Xie was supported by NSFC 61772094, China.

978-3-9819263-4-7/DATE20/©2020 EDAA

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.

contributions can be summarized as follows.

- We propose a high frequency electro-optical CNN accelerator, LightBulb, to accelerate the inferences of binarized CNNs without significant accuracy degradation. We present a microdisk-based XNOR gate and a photonic PCM-based ADC for LightBulb to process the XNORs and popcounts of binarized CNNs.
- We further integrate photonic racetrack memory into LightBulb to serve as input/output registers. Since the read and write are two critical steps of the LightBulb pipeline, the access latency of input/output registers decides its pipeline operating frequency. Photonic racetrack memory enables LightBulb to work at 50GHz.
- We implement, evaluate and compare LightBulb and against the state-of-the-art GPU, FPGA, ASIC, ReRAM and photonic CNN accelerators. Our experimental results show LightBulb improves the inference throughput by $17\times \sim 173\times$ and the throughput per Watt by $17.5\times \sim 660\times$ over various prior CNN accelerators.

II. Background, Related Work and Motivation

A. CNN Basics and Network Binarization

Inference latency matters. The state-of-the-art CNNs [12], [10], [6] show unreasonable effectiveness in various applications such as natural language processing, computer vision, and recommender systems. Although a CNN model is trained, billions of **inference**s will be performed on the model. Particularly, edge devices send their inference tasks to data centers [8] and rely on the computing power of data centers to support their real-time CNN applications. Therefore, the CNN inference latency in data centers matters. In contrast, CNN trainings are done offline. In this paper, we focus on accelerating CNN inferences in data centers.

Inference overhead is large. A CNN typically comprises multiple layers including convolutional, fully-connected, batch normalization, activation and pooling layers. In a CNN, the convolutional layers consume $> 90\%$ of its inference latency [31]. A convolutional layer receives N feature matrices as inputs, each of which is convolved by a moving window with a $K \times K$ weight kernel to produce an element in one of output feature matrices. The stride of the moving window is represented by S, which is often smaller than K. The output consisting of M feature matrices is the input for the next convolutions layer. An CNN inference [6] involves billions of floating-point multiply-accumulate (MAC) operations. Because of the large computing overhead, the battery of Google Glass stands for only 45 minutes [5] when tracking consecutive human actions. These edge devices have to rely on data centers to process computationally expensive CNNs.

Network binarization. To reduce the computing overhead of CNN inferences, network binarization techniques [21], [15], [32] are proposed to approximate floating-point inputs, weights and activations of a CNN by the linear combination of multiple binary filters constrained to -1 or +1, so that computationally expensive floating-point MACs can be replaced by bitwise XNOR and popcount operations. For example, for a convolutional layer with N input channels and M output channels, its weight $\mathbf{W} \in \mathbb{R}^{K\times K\times N\times M}$ is approximated by P binary filters $\mathbf{B}_0, \ldots, \mathbf{B}_{P-1} \in \{-1. + 1\}^{K\times K\times N\times M}$, such that $\mathbf{W} \approx \psi_0\mathbf{B}_0 + \psi_1\mathbf{B}_1 + \ldots + \psi_{P-1}\mathbf{B}_{P-1}$, where K is the weight kernel size, and ψ is the weight scaling factor vector. In the same way, its input $\mathbf{I} \in \mathbb{R}^{L\times H\times N}$ can be approximated by Q binary filters $\mathbf{B}_0, \ldots, \mathbf{B}_{Q-1} \in \{-1. + 1\}^{L\times H\times N}$, such that $\mathbf{I} \approx \lambda_0\mathbf{B}_0 + \lambda_1\mathbf{B}_1 + \ldots + \lambda_{Q-1}\mathbf{B}_{Q-1}$, where L is the input length, H is the input height, and λ is the input scaling factor vector. So the convolution can be approximated by

$$Conv(\mathbf{W}, \mathbf{I}) \approx \sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} \psi_p\lambda_q Conv(\mathbf{B}_p, \mathbf{B}_q) \qquad (1)$$

, where the binary convolution $Conv(\mathbf{B}_p, \mathbf{B}_q)$ can be computed by XNOR and popcount operations.

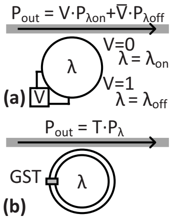

$P_{out} = V\cdot P_{\lambda on}+\overline{V}\cdot P_{\lambda off}$
V=0
$\lambda = \lambda_{on}$
V=1
$\lambda = \lambda_{off}$
λ
V
(a)
$P_{out} = T\cdot P_{\lambda}$
GST
λ
(b)

Fig. 1. Microdisk and pPCM.

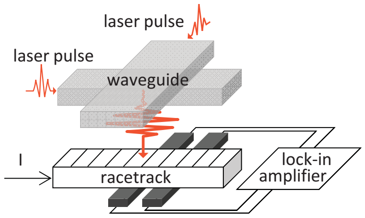


Fig. 2. Photonic racetrack memory.

B. Photonic Devices

Silicon microdisk. The silicon microdisk technology [29], [30] emerges as one of the most promising solutions to future photonic computing, due to its CMOS compatibility, small footprint, i.e., $25\mu m^2$, and ultra-low-power consumption, i.e., $1fJ/bit$, as shown in Table I. As Figure 1(a) shows, the wavelength of a microdisk resonator λ can be controlled by a *forward-bias* voltage V. If $V = 0$, $\lambda = \lambda_{off}$. Only the input power with λ_{off}, $P_{\lambda_{off}}$, can pass through the waveguide, so the output power $P_{out} = P_{\lambda_{off}}$. If $V = 1$, $\lambda = \lambda_{on}$. The output power $P_{out} = P_{\lambda_{on}}$. If both $P_{\lambda_{off}}$ and $P_{\lambda_{on}}$ are sent to the waveguide, the output power is decided by the forward-bias voltage V. We have $P_{out} = V \cdot P_{\lambda_{on}} + \overline{V} \cdot P_{\lambda_{off}}$.

TABLE I
The comparison between SRAM and photonic devices.

	SRAM	microdisk	pRacetrack	pPCM
area (μm^2)	0.15	8.03	100	9.07
bit/cell	1	1	128	5
energy (fJ/bit)	1100	1	10	50
frequency (GHz)	5	50	100	50

Photonic Phase Change Memory. Recent works [1], [22], [14] adopt photonic Phase Change Memory (pPCM) to accelerate floating-point multiplications. As Figure 1(b) shows, a small block of phase change material GST is fabricated on a microdisk. By controlling the GST element, a microdisk can achieve different transmissions. If the GST element is in the amorphous state, the microdisk has high transmission. In contrast, if the GST element is in the crystalline state, the microdisk obtains low transmission. A pPCM cell can store 5 bits [14]. The output power P_{out} of the pPCM-based microdisk can be represented by $T \cdot P_\lambda$, where T is the transmission of the pPCM, and P_λ indicates the input power.

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.

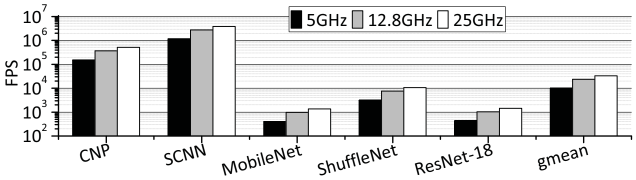


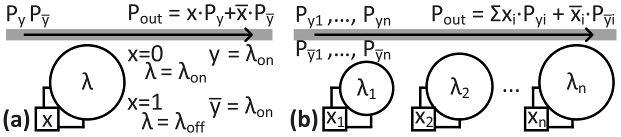
Fig. 3. The inference throughput of HolyLight at various frequencies (FPS: frames/images per second).

Photonic Racetrack Memories. To implement high density ultra-fast photonic memory, a recent work [11] creates photonic racetrack (pRacetrack) memory that can be read and written by light. Figure 2 shows an example of a pRacetrack cell whose major component is a racetrack structure consisting of multiple magnetic domains separated by domain walls. Each domain has its own magnetization direction. The data can be represented by the magnetization direction of each domain. A racetrack structure can contain 128 domains [28]. All domains in a cell share several ports for reads and writes. The motion of magnetic domain walls in a racetrack structure can be controlled by a electrical pulse on the head of the racetrack structure. Unlike the electrical racetrack memory, a pRacetrack cell has two waveguides orthogonally intersected on its access port. By simultaneously injecting two light pulses, a strengthened vertical light switches the magnetization direction of the target domain. To access a domain, we first shift the target domain to the access port and then write or read the domain by light pulses. A write on a pRacetrack cell costs only $10ps$ [11]. Two wires connected to a lock-in amplifier form a Hall cross. During a read, when the light pulse is applied on a domain, the lock-in amplifier can measure the out-of-plane magnetization by the anomalous Hall effect.

C. Related Work and Motivation

Although recent works [26], [16] build all-optical accelerators to process convolutional, activation, fully-connected layers of a CNN, their inference accuracy is seriously limited by the errors caused by imperfections of photonic devices. For example, a recent all-optical accelerator [16] achieves only 91.75% inference accuracy on the small hand-written digit dataset MNIST. The imperfections of photonic devices completely offset the benefits of a CNN, since even the traditional support vector machine algorithm [12] can obtain $> 98\%$ inference accuracy.

Prior work [17] presents an electro-optical CNN accelerator, HolyLight, that uses photonic matrix-vector multipliers to accelerate convolutions and electrical components to perform activations to achieve originally high inference accuracy. However, HolyLight suffers from low frequency, due to the slow SRAM registers and the long critical path of its ripple-carry adders. HolyLight integrates SRAM-based input/output registers into its pipeline, so the SRAM latency decides its operating frequency. Although HolyLight claims that it can run at $12.8GHz$, the fastest SRAM arrays run at only $5GHz$. As Figure 3 shows, if we consider the low SRAM frequency ($5HGz$), on average, the CNN inference throughput of HolyLight degrades by 58%. If we have ideal photonic registers and adders enabling HolyLight to work at $25GHz$, the inference throughput of HolyLight improves by 40% averagely. Higher frequency significantly improves the inference throughput of the electro-optical accelerator.



P_y $P_{\overline{y}}$
$P_{out} = x \cdot P_y + \overline{x} \cdot P_{\overline{y}}$
x=0
$\lambda = \lambda_{on}$
$y = \lambda_{on}$
λ
x=1
$\lambda = \lambda_{off}$
$\overline{y} = \lambda_{on}$
(a)
x
$P_{y1}, \ldots, P_{yn}$
$P_{\overline{y}1}, \ldots, P_{\overline{y}n}$
$P_{out} = \Sigma x_i \cdot P_{yi} + \overline{x}_i \cdot P_{\overline{y}i}$
λ_1
λ_2
...
λ_n
(b)
x_1
x_2
x_n

Fig. 4. Photonic XNOR gate: (a) a 1-bit gate and (b) an n-bit unit.

III. LightBulb

To avoid the ripple-carry adders having a long critical path, we propose a novel electro-optical CNN accelerator, LightBulb, to accelerate the XNOR and popcount operations of binarized CNNs using photonic microdisk-based XNOR gates and pPCM-based ADCs. We also integrate photonic racetrack memory as the input/output registers to enhance the operating frequency of LightBulb. At last, we further develop a pipelined LightBulb design to maximize the CNN inference throughput.

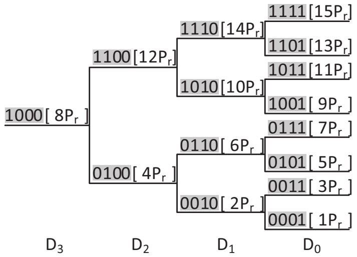


Fig. 5. The binary search tree for a 4-bit ADC.

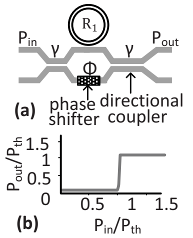


Fig. 6. A pComparator.

A. Binary Convolution: XNOR-Popcount

1) Photonic XNOR unit: We propose a photonic XNOR gate to compute XNORs by a single microdisk. As Figure 4(a) exhibits, we uses the bias voltage V of the microdisk to serve as one input x of the XNOR gate. The input power of the microdisk is used as the other input y of the XNOR gate. Particularly, the input power at λ_{on} indicates y, while the input power at λ_{off} represents $\overline{y}$. If $x = 0$, the bias voltage is 0 and the microdisk works at λ_{off}, so the output power of the microdisk P_{out} equals to $\overline{x} \cdot \overline{y}$. If $x = 1$, the bias voltage is 1 and the microdisk is operated at λ_{on}, so the output power of the microdisk P_{out} equals to $x \cdot y$. In short, $P_{out} = x \cdot y + \overline{x} \cdot \overline{y}$. To compute multiple XNOR operations, we presents an n-bit XNOR unit consisting of n 1-bit photonic XNOR gates connected to the same waveguide. These photonic XNOR gates can work simultaneously by wavelength-division multiplexing (WDM), because the microdisks of the XNOR gate from the left to the right in Figure 4(b) have enlarging radiuses and hence they are operated at different wavelengths. To avoid interferences, we conservatively connect only 16 XNOR gates having 16 different λ_{on}s and 16 different λ_{off}s to one shared waveguide. A 16-bit XNOR unit generates the analog popcount value of the sum of 16 XNOR operations as its output power.

2) pPCM-based ADC: An ADC is required to convert the analog popcount value of the sum of 16 XNOR operations to a digital value. However, the CMOS ADC has become the

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.

largest performance and power bottleneck in prior ReRAM-based [13] and photonic [17] CNN accelerators, due to its large area overhead and huge power consumption. In this paper, we propose an all-optical ADC to accelerate analog-to-digital conversions by pPCMs. We use pPCMs to implement the Hopfield network [3] for the temporal binary search of an ADC from the most significant bit (MSB) to the least significant bit (LSB). The temporal binary search of a 4-bit ADC example is shown in Figure 5 and can also be described as follows:

$$\begin{aligned} D_3 &= Neuron(P_{in} - 8P_r) \\ D_2 &= Neuron(P_{in} - 4P_r - 8P_{D_3}) \\ D_1 &= Neuron(P_{in} - 2P_r - 4P_{D_2} - 8P_{D_3}) \\ D_0 &= Neuron(P_{in} - P_r - 2P_{D_1} - 4P_{D_2} - 8P_{D_3}) \end{aligned} \tag{2}$$

, where P_{in} is the analog input; $D_3 \sim D_0$ means its digital value (1 or 0); $Neuron$ denotes a neuron that outputs 1 if its input > 0, otherwise it produces 0; and P_r is a reference power equal to the smallest discrete power quantum. Since D_x is 1 or 0, we have $\overline{D_x} = 1 - D_x$, where $0 \leq x \leq 3$. And if $D_x = 1$, we define the input power $P_{D_x} = P_r$; otherwise $P_{D_x} = 0$. As Equation 2 suggests, computing each digital bit of the ADC is sequential from D_3 to D_0. This process can be described as searching the binary tree with all possible combinations. Since we cannot deal with subtractions inside the ADC, we equivalently transform Equation 2 to

$$\begin{aligned} D_3 &= Co(P_{in}, 8P_r) \\ D_2 &= Co(P_{in} + 8P_{\bar{D}_3}, 12P_r) \\ D_1 &= Co(P_{in} + 4P_{\bar{D}_2} + 8P_{\bar{D}_3}, 14P_r) \\ D_0 &= Co(P_{in} + 2P_{\bar{D}_1} + 4P_{\bar{D}_2} + 8P_{\bar{D}_3}, 15P_r) \end{aligned} \tag{3}$$

, where the function Co compares its two inputs and outputs 1 if the first input is larger. To compute the MAC operations of the first input, as Figure 7 shows, we employ a row of pPCM cells, each of which works at different wavelengths. The transmissions of the pPCM cells represent the constant parameters of each item in Equation 3. P_{in} and $P_{\overline{D_x}}$ are multiplexed and transmitted the pPCM row, so that a pPCM cell with wavelength $\lambda_{\overline{D_x}}$ can only reacts on the input power $P_{\overline{D_x}}$ at the same wavelength. Its output power would be $T_{yx} \cdot P_{\overline{D_x}}$, where T_{yx} is the transmission of the xth pPCM cell for computing D_y. The com function sums output power of pPCM cells ($P_{in} + \sum T_{yx} \cdot P_{\overline{D_x}}$) and compares the summed power against a constant threshold. For instance, to compute D_1, in Figure 7(c), the transmissions of the pPCM cells, T_{12} and T_{13}, are used to represent 4 and 8. P_{in}, $P_{\bar{D}_2}$ and $P_{\bar{D}_3}$ are multiplexed and inputted into the pPCM array, so that $P_{in} + 4P_{\bar{D}_2} + 8P_{\bar{D}_3}$ is computed by the Co function that will also compare it against $14P_r$. We use a broadband photonic comparator (pComparator) [9] to implement the Co function. The pComparator consisting of a phase shifter and a directional coupler can be seen in Figure 6. If the input power of a pComparator is larger than its threshold, the pComparator outputs 1, otherwise it generates 0. We set the first input of the Co function as the pComparator input, and the second input of the Co function as the pComparator threshold. During an ADC operation, we do not need to write the pPCM cell, so the pPCM-based ADC has no worn-out issue.

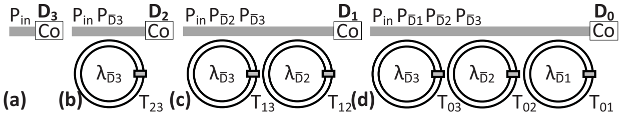


Fig. 7. A 4-bit pPCM-based ADC.

B. Binarization, Batch Normalization, Activation, and Pooling

1) Binarization: We use CMOS circuits to binarize the inputs of each layer by the LQ-Nets binarization [32], while the weights of each layer are binarized during the training.

2) Batch Normalization: Batch normalization layers [21] significantly enhance the CNN inference accuracy. Each input x is normalized through $\frac{x-\mu}{\sqrt{\sigma^2+\epsilon}}\gamma + \beta$, where μ is the mean of the batch, γ is the scale, β is the shift, and σ^2 is the variance with a tiny constant ϵ to void zero denominator. Except the input x, all the other variables are obtained during the training. We use two 16-bit photonic adders and a 16-bit photonic multiplier to perform the batch normalization.

3) Activation: We adopt electrical activation units to support various types of activation functions including $ReLU$, $sigmoid$ and $tanh$.

4) Pooling: We use 16-bit electrical comparators to implement max pooling units by linearly scanning the data stream and always keeping the latest maximum value. For an $n:1$ max pooling, the latest maximum value is produced and reset every n cycles. An electrical average pooling unit consists of a 16-bit adder and shifter registers. For an $n:1$ average pooling, the average result is generated every n cycles.

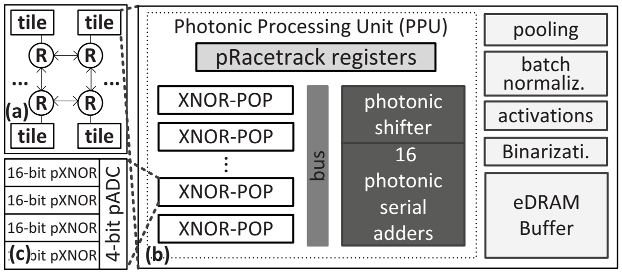


Fig. 8. The overall architecture of LightBulb.

C. Pipelined LightBulb

1) Architecture: The overall architecture of LightBulb is shown in Figure 8(a), where the LightBulb chip consists of multiple tiles connected by routers and a network-on-chip. Each tile communicates with the other tiles via its router. As Figure 8(b) illustrates, each tile includes a photonic processing unit (PPU) and electrical peripheral circuits. A PPU uses photonic XNOR-POP units to accelerate XNOR and popcount operations. In Figure 8(c), an XNOR-POP unit is composed of four 16-bit photonic XNOR units and one 4-bit pPCM ADC. Photonic shifters and adders are employed by a PPU to support factor scaling and aggregate intermediate results. All photonic computing devices adopt pRacetrack-based input/output registers to achieve $50GHz$ operating frequency. Electrical peripheral circuits are designed to process pooling, batch normalization, binarization, activations.

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.

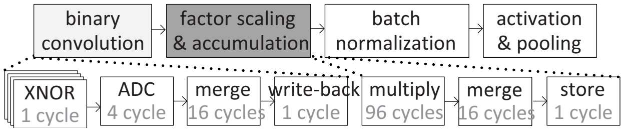


Fig. 9. The LightBulb pipeline.

2) Pipeline: The pipeline of LightBulb is described in Figure 9. To compute Equation 1, a PPU of LightBulb first performs binary convolutions (XNOR-POPs), factor scaling operations and accumulations. It then applies batch normalization, activation and pooling operations on the results of Equation 1 by electrical circuits, since these operations are not the performance bottleneck. We design a pipeline for the binary convolutions, since it costs the largest portion of the inference time. Four 16-bit photonic XNOR units work with a 4-bit pPCM-based ADC in a PPU. During each cycle, the PPU produce a 4-bit digital XNOR-POP value. It adopts 16 photonic serial adders to merge the digital XNOR-POP values to a 16-bit output of the binary convolution, which will be written back to pRacetrack-based registers at the end of the pipeline. After registers are full, the PPU stops binary convolutions, starts to use all serial adders and a 16-bit shifter to conduct factor scaling and accumulation operations, and writes the final result back to pRacetrack-based registers

TABLE II
THE HARDWARE COST OF LIGHTBULB.

name	component	spec	power(mW)	Area(mm^2)
XNOR-POP	16-bit XNOR	×4	3.2	0.000514
	4-bit ADC	×1	0.2	0.00017
sub-total			3.4	0.00069
50GHz PPU	XNOR-POP	×25	85	0.0172
	16 serial pAdder	×25	1060	0.0364
	16-bit pShifter	×25	5.12	0.16
	pRacetrack reg.	2.25KB	200	0.0143
sub-total			1350.12	0.228
Tile	PPU	×1	1350.12	0.228
	activation	×1	0.26	0.0003
	binarization	×1	0.18	0.0002
	pooling	×1	0.4	0.000224
	eDRAM	128KB	31.2	0.134
	bus	384-wire	7	0.009
	router	32-flit, 8-port	42	0.151
sub-total			1431.16	0.5227
total	tile	×46	65.83W	24.05
HolyLight			68.3W	22.46

D. Design Overhead

The power and area overhead of LightBulb is exhibited in Table II. To compare against a prior photonic CNN accelerator HolyLight [17], we adopted its electrical activation and pooling units, eDRAM buffers, bus and routers. The CMOS circuits are synthesized by Cadence Virtuoso with 32nm process technology. To simulate our photonic microdisk-based and pPCM-based components, we used Lumerical FDTD [18] and INTERCONNECT [19]. We modified CACTI to model the power and area of pRacetrack-memory-based registers. We also adopted the optical splitters & combiners, optical multiplexers, photodetectors and microdisks from HolyLight [17].

IV. EXPERIMENT METHODOLOGY

Workload. The CNNs we simulated are listed in Table III. We binarized the weights and activations of all CNNs with 3-bit by the LQ-Nets binarization technique [32]. CNP [4] and SCNN [27] were trained with MNIST to recognize simple hand-written digits, while MobileNetV2 [24], ShuffleNetV2 [20] and ResNet-18 were trained with ImageNet to classify complex objects. We trained all CNNs by Tensorflow. We also compare the inference accuracy of full precision and binarized CNNs. Binarized CNNs degrade the inference accuracy of MNIST CNNs by $0.1\% \sim 0.5\%$ and decreases the top5% inference accuracy of ImageNet CNNs by $1 \sim 2\%$.

TABLE III
THE CNN BENCHMARKS (ACC: ACCURACY; C: CONVOLUTIONAL LAYER; P: POOLING LAYER; F: FULLY CONNECTED LAYER).

name	database	topology	$acc_{orig}(\%)$	$acc_{bin}(\%)$
CNP	MNIST	3C,2P,1F	97.0	96.7
SCNN	MNIST	2C,2F	99.0	98.2
MobileNet	ImageNet	10C,1P,1F	92.5	91.4
ShuffleNet	ImageNet	5C,2P,1F	88.4	87.3
ResNet-18	ImageNet	18C,2P,1F	89.2	87.9

Scheme. We compared LightBulb against six counterparts shown in Table IV. We selected an Nvidia Tesla GPU, a Xilinx Virtex7 FPGA [31], two ASIC chips including DaDianNao [2] and Google TPU [7], a ReRAM-based PIM ISAAC [25] and a photonic CNN accelerator [17]. Unlike the single chip DaDianNao, Google TPU comprises four chips, each of which can achieve larger throughput but consume more power. ISAAC accelerates convolutions by ReRAM dot-product engines. HolyLight quantizes weights into the power-of-2 representations, so that it can compute convolutions by additions and shifts. It uses photonic ripple-carry adders and shifters to accelerate additions and shifts.

TABLE IV
THE SCHEME COMPARISON (NORMALIZED TO $32nm$).

Name	Description	$Power\ (W)$
GPU	Nvidia Tesla P100	250
FPGA	Xilinx Virtex7 VX485T	40
DaDianNao	ASIC	20.1
TPU	4-chip ASIC	384
ISAAC	ReRAM PIM	65.8
HolyLight	photonic accelerator	68.3

Simulation. We used the deep learning accelerator simulator FODLAM [23] to evaluate the inference throughput, power and energy consumption of all accelerators. We implemented the pipeline details of LightBulb into FODLAM. Through an accelerator configuration and a network description, FODLAM generates details on the throughput and power of each accelerator executing the network.

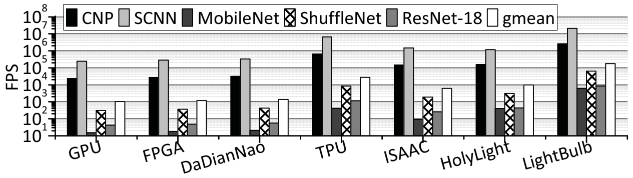


Fig. 10. The inference throughput of various hardware platforms (FPS: frames/images per second).

V. EVALUATION

Inference throughput. The FPS comparison of all accelerators is shown in Figure 10. Among all prior designs, Google

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.

TPU obtains the best throughput. On average, compared to photonic HolyLight, Google TPU improves CNN inference throughput by $1.77\times$, since it has 4 chips and costs $5.6\times$ power consumption. Compared to the other prior CNN accelerator designs, HolyLight enhances inference throughput by $60\% \sim 7.7\times$, due to its power-of-2 quantization and power-efficient photonic devices. Our LightBulb improves the inference throughput by $16.9\times$ and $5.4\times$ over HolyLight and TPU, respectively. Because the CNN binarization technique LightBulb adopts achieves similar inference accuracy with much smaller computing overhead, when compared to the full precision and the power-of-2 quantized CNNs. The photonic XNOR and pPCM-based ADC units in LightBulb work at much higher frequency than the ripple-carry adders in HolyLight. The pRacetrack-based registers further enable the entire LightBulb accelerator to operate at $50GHz$. On the contrary, the SRAM-based registers limit the frequency of HolyLight to only $5GHz$.

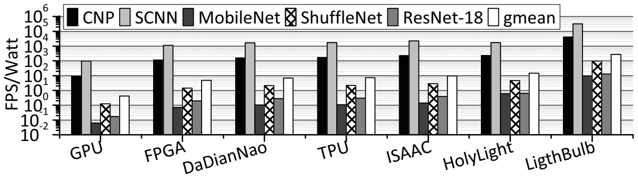


Fig. 11. The inference throughput per Watt of various hardware platforms (FPS: frames/images per second).

Throughput per Watt. The comparison on FPS per Watt of all accelerators is shown in Figure 11. When considering the power consumption, FPGA obtains $11\times$ inference energy efficiency over GPU, while ASIC designs including DaDianNao and Google TPU improve the FPS per Watt by 45% and 52% respectively over FPGA. The ReRAM-based PIM ISAAC further enhances the FPS per Watt by only 30% over Google TPU, mainly because the power-hungry ADCs become the power bottleneck of ISAAC. Compared to prior non-photonic CNN accelerators, HolyLight boosts the inference throughput per Watt by $54\% \sim 34\times$, due to its power-efficient photonic adders and shifters. Our LightBulb improves the inference throughput per Watt by $17.5\times$ over HolyLight. The network binarization technique of LightBulb greatly reduces the essential computing effort of CNN inferences. Moreover, the photonic XNOR gates, pPCM-based ADCs and pRacetrack-based registers make LightBulb work at $50GHz$ frequency.

VI. Conclusion

In this paper, we propose an electro-optical CNN accelerator, LightBulb, to process the inferences of binarized CNNs by photonic XNOR and popcount units. We first binarize the weights and activations of CNNs into the linear combinations of multiple {-1,+1}s, so that floating-point MAC operations can be replaced by XNORs and popcounts. We then presents microdisk-based XNOR units and pPCM-based ADCs to accelerate XNORs and popcounts. Compared to prior CNN accelerators, LightBulb improves the inference throughput by $17\times \sim 173\times$ and the inference throughput per Watt by $17.5\times \sim 660\times$ with $\sim 2\%$ accuracy degradation.

References

[1] I. Chakraborty, *et al.*, “Toward fast neural computing using all-photonic phase change spiking neurons,” *Scientific reports*, 8(1), 2018.

[2] Y. Chen, *et al.*, “DaDianNao: A Machine-Learning Supercomputer,” in *MICRO*, 2014.

[3] L. Danial, *et al.*, “Breaking Through the Speed-Power-Accuracy Tradeoff in ADCs Using a Memristive Neuromorphic Architecture,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(5):396–409, 2018.

[4] C. Farabet, *et al.*, “CNP: An FPGA-based processor for Convolutional Networks,” in *FPL*, 2009.

[5] W. Glauser, “Doctors among early adopters of Google Glass,” *Canadian Medical Association Journal*, 185(16):1385, 2013.

[6] K. He, *et al.*, “Deep Residual Learning for Image Recognition,” in *CVPR*, 2016.

[7] N. P. Jouppi, *et al.*, “In-datacenter performance analysis of a tensor processing unit,” in *ISCA*, 2017.

[8] Y. Kang, *et al.*, “Neurosurgeon: Collaborative Intelligence Between the Cloud and Mobile Edge,” in *ASPLOS*, 2017.

[9] H. Kishikawa, *et al.*, “Optical thresholder consisting of two cascaded Mach-Zehnder interferometers with nonlinear microring resonators,” *Optical Engineering*, 56(8):086101, 2017.

[10] A. Krizhevsky, *et al.*, “ImageNet Classification with Deep Convolutional Neural Networks,” in *NIPS*, 2012.

[11] M. L. Lalieu, *et al.*, “Integrating all-optical switching with spintronics,” *Nature communications*, 10(1):110, 2019.

[12] Y. Lecun, *et al.*, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, 86(11), Nov 1998.

[13] B. Li, *et al.*, “MErging the Interface: Power, area and accuracy co-optimization for RRAM crossbar-based mixed-signal computing system,” in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, 2015.

[14] X. Li, *et al.*, “Fast and reliable storage using a 5 bit, nonvolatile photonic memory cell,” *Optica*, 6(1):1–6, 2019.

[15] X. Lin, *et al.*, “Towards accurate binary convolutional neural network,” in *Advances in Neural Information Processing Systems*, pages 345–353, 2017.

[16] X. Lin, *et al.*, “All-optical machine learning using diffractive deep neural networks,” *Science*, 361(6406):1004–1008, 2018.

[17] W. Liu, *et al.*, “HolyLight: A Nanophotonic Accelerator for Deep Learning in Data Centers,” in *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1483–1488, 2019.

[18] Lumerical, “FDTD Solutions,” `http://www.lumerical.com/tcad-products/fdtd/`.

[19] Lumerical, “INTERCONNECT,” `http://www.lumerical.com/tcad-products/interconnect/`.

[20] N. Ma, *et al.*, “ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design,” in *ECCV*, 2018.

[21] M. Rastegari, *et al.*, “XNOR-Net: Imagenet Classification Using Binary Convolutional Neural Networks,” in *ECCV*, 2016.

[22] C. Ríos, *et al.*, “Integrated all-photonic non-volatile multi-level memory,” *Nature Photonics*, 9(11):725, 2015.

[23] A. Sampson and M. Buckler, “FODLAM: a first-order deep learning accelerator model,” `https://github.com/cucapra/fodlam`.

[24] M. Sandler, *et al.*, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in *CVPR*, 2018.

[25] A. Shafiee, *et al.*, “ISAAC: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars,” in *ISCA*, 2016.

[26] Y. Shen, *et al.*, “Deep learning with coherent nanophotonic circuits,” *Nature Photonics*, 2017.

[27] P. Y. Simard, *et al.*, “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis,” in *ICDAR*, 2003.

[28] Z. Sun, *et al.*, “Cross-layer racetrack memory design for ultra high density and low power consumption,” in *DAC*, 2013.

[29] Z. Ying, *et al.*, “Electro-Optic Ripple-Carry Adder in Integrated Silicon Photonics for Optical Computing,” *IEEE Journal of Selected Topics in Quantum Electronics*, 2018.

[30] Z. Ying, *et al.*, “Silicon microdisk-based full adders for optical computing,” *Optics letters*, 43(5), 2018.

[31] C. Zhang, *et al.*, “Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks,” in *FPGA*, 2015.

[32] D. Zhang, *et al.*, “Lq-nets: Learned quantization for highly accurate and compact deep neural networks,” in *ECCV*, 2018.

Authorized licensed use limited to: Indiana University. Downloaded on July 03,2020 at 13:18:07 UTC from IEEE Xplore. Restrictions apply.