A Winograd-Based Integrated Photonics Accelerator for Convolutional Neural Networks

Armin Mehrabian, *Member, IEEE*, Mario Miscuglio, Yousra Alkabani, *Member, IEEE*, Volker J. Sorger, *Senior Member, IEEE*, and Tarek El-Ghazawi, *Fellow, IEEE*

(Invited Paper)

***Abstract*—Neural Networks (NNs) have become the mainstream technology in the artificial intelligence (AI) renaissance over the past decade. Among different types of neural networks, convolutional neural networks (CNNs) have been widely adopted as they have achieved leading results in many fields such as computer vision and speech recognition. This success in part is due to the widespread availability of capable underlying hardware platforms. In parallel, hardware specialization can expose us to novel architectural solutions, which can outperform general purpose computers for the tasks at hand. Although different applications demand for different performance measures, they all share speed and energy efficiency as high priorities. Meanwhile, photonics processing has seen a resurgence due to its inherited high speed and low power nature. Here, we investigate the potential of using photonics in CNNs by proposing a CNN accelerator design based on Winograd filtering algorithm. Our evaluation results show that while a photonic accelerator can compete with current state-of-the-art electronic platforms in terms of both speed and power, it has the potential to improve the energy efficiency by up to three orders of magnitude.**

***Index Terms*—Convolutional neural networks, photonics, Winograd.**

I. Introduction

THE field of AI has undergone revolutionary progress over the past decade. Invited Wide availability of data and cheaper than ever compute resources have contributed immensely to this growth. At the same time, advancements is the field of modern neural networks, known as deep learning (DL) have attracted the attention of academia and industry. This popularity is mainly owed to neural networks' success in a large gamut of AI applications including but not limited to computer vision, speech recognition, and natural language processing. Among the different types of neural networks, CNNs are considered the most viable architecture for AI applications. CNNs are remarkably versatile in most AI tasks. However, all of this comes at the price of high computational costs.

In the meantime, the use of integrated photonics in neural networks for implementing neuron functionalities has shaped to be an attainable alternative near future technology for limiting the power consumption and increasing the operating speed [1]–[3]. Photonics benefit from the coherent nature of electromagnetic waves, which interfere while propagate through a photonic integrated circuit (PIC). Central to many AI techniques and algorithms is implementation of hardware solutions that mimic the multiply and accumulate (MAC) function. The main advantage of photonic neural networks over electronics is that the energy consumption for performing a series of multiplications and additions does not scale with MAC speed. The training of an optical neural network necessitates an active modulation of the optical signal in a hybrid opticalelectronic configuration [4]. For this reason, these architectures face significant hurdles when compared to their electronic counterparts. To be competitive, they are expected to have low power consumption and high-speed electro-optic modulation [5]–[7]. Additionally, they require to pair with electrical to optical (EO), optical to electrical (OE) converters, and I/O interfaces. However, when trained, photonic neural networks do not rely on any additional energy for active switching. Therefore the architectures that perform tasks such as weighting, can be realized completely passive, and the computations happen without the consumption of any dynamic power [8]–[10]. In this panorama, all-optical neural networks (AONNs) represent a promising future. Current all-optical implementations in free space [11] and in integrated photonics [12]–[14] can outperform their electronic counterparts providing promises of great energy efficiency and speed enhancement for learning tasks.

In this manuscript, we explore the potentials of using high-speed, low-power photonics in a CNN accelerator by exploiting coherent all-optical matrix multiplication in wavelength division multiplexing (WDM), using microring resonator weight banks (MRRs). Our architecture is inspired by [15], [16], where Winograd filtering algorithm is adopted to perform convolution to speedup the execution time and reduce the computational complexity. We investigate the performance of our proposed architecture in terms of speed and power. Since our proposed

Manuscript received April 1, 2019; revised November 4, 2019; accepted November 29, 2019. Date of publication December 3, 2019; date of current version December 19, 2019. The work of V. J. Sorger and T. El-Ghazawi was supported by the Office of Navy Research (ONR) under Award N00014-19-1-2595 of the Electronic Warfare program. *(Corresponding author: Armin Mehrabian.)*

The authors are with the Department of Electrical and Computer Engineering, George Washington University, Washington, DC 20052 USA (e-mail: armin@gwu.edu; mmiscuglio@gwu.edu; yousra@gwu.edu; sorger@gwu.edu; tarek@gwu.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTQE.2019.2957443

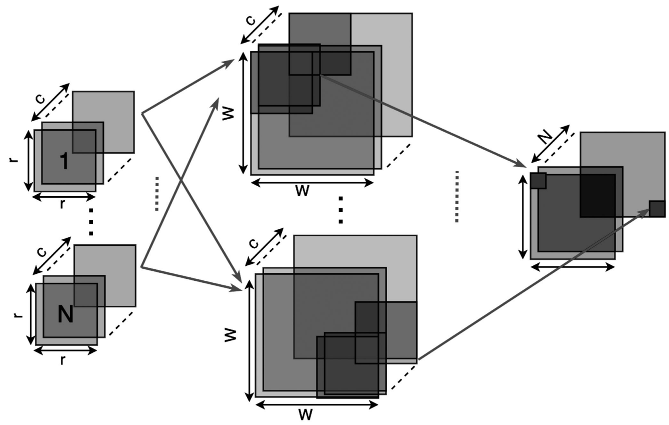

Fig. 1. A single layer of a CNN. Each of the N filters (left) scan the input feature maps (middle) for features. This results in output feature maps, with N channels equal to the total number of filters.

architecture is analog at the core, we also investigate the robustness of neural networks executed on our proposed design in terms of tolerance against noise.

We summarize the main contributions of this work as,

- a first proposed photonic CNN architecture based on the Winograd filtering algorithm
- an analytical framework to evaluate the speed performance of our proposed accelerator
- an in-house simulator based on a modified Google Tensorflow tool to simulate the performance of our proposed photonic accelerator with power and noise awareness
- a modified training process to enhance robustness to inevitable hardware noise sources during the inference stage

II. Convolutional Neural Networks (CNNs)

A CNN is a neural network comprised of one or more convolutional layers. CNNs are mostly known for their great performance on image data, however, their application extend to many other data types with local features. At the very high level, each convolution layer uses a collection of feature detectors, known as filters, that scan input data for presence or absence of a particular set of features. Hence, in a CNN layer, inputs and outputs are referred to as feature maps (fmap). By cascading multiple of these convolutional layers, a hierarchy of feature detectors are formed. In this hierarchy, feature detectors closer to the input detect primitive features. As we move towards the final layers, the type of features detected become more abstract. Conventionally, the dimension of each filter in a a CNN is $3D$ with the two first dimensions being the height and width of the filer and the last dimension, known as the channel dimension, represents various filters. The use of convolutional filters to scan input data had been practiced well before the rise of the field of deep learning and CNNs. However, in traditional signal processing, such filters are hand-engineered by experts, which can be costly, only designed for specific purposes, and vulnerable to designer bias. In a modern CNN, these filters are learned through the training process. Figure 1 shows the overall architecture of a CNN layer.

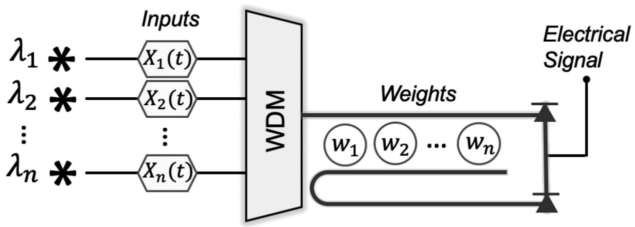

Fig. 2. A broadcast-and-weight neuron. Inputs X_i modulate different wavelength lasers. Modulated beams are then bundled through WDM.

III. Photonic Realization of CNNs

In data communication and computation, photonics has the potential to offer practical solutions to overcome some of the limitations currently facing electronic systems. In a neuromorphic system, processing elements (PEs) are arranged in a distributed fashion with ideally large number of incoming (fan-in) and outgoing (fan-out) connections. Inspired by biological neural systems, some of these connection are required to connect neurons across farthest parts of the brain. In addition, neuromorphic PEs are in large part specific-purpose processors in contrast to the general purpose processors.

Neuromorphic processing can benefit from photonics in three major ways. First, photonics can significantly reduce the amount of energy consumed in interconnects among PEs by avoiding energy dissipation due to charging and discharging of electrical wires. Secondly, current neuromorphic algorithms known to neural networks, and in particular in CNNs, heavily rely on the multiply and accumulate (MAC) operation, which can be realized with very low energy budgets in photonics. Thirdly, photonics can increase communication and computation bandwidth by exploiting WDM. The adoption of WDM allow for higher density of computation and communication between PEs by packing more channels and parallel computations in a neuromorphic processor.

A. Photonic Convolution Kernels and MAC Operation

One major advantage of a photonic MAC operation is that it can be performed with almost zero energy [17]. However, if the signal is converted from optical to electrical, the conversion and successive electronic manipulations impose additional energy loss. To build a photonic convolutional filter, we use a microring resonator (MRR) network proposed in [18]. Figure 2 depicts a single MRR neuron. In this schema input WDM signals are weighted through tunable MRRs. These weighted inputs are later incoherently summed up using a photodetector, which amounts to a MAC operation.

Thus, by the use of N wavelengths, it is possible to establish up to N^2 independent connections. Maximum N with current technologies is estimated to be around 108 channels resulting in a total of 10 k connections [19]. It should be note that having closely spaced wavelengths as multiple laser sources while tuning the rings to match both resonance and FSR is a very challenging task. Although, on the source side, a set of phase-locked, equally spaced laser frequency lines can be generated

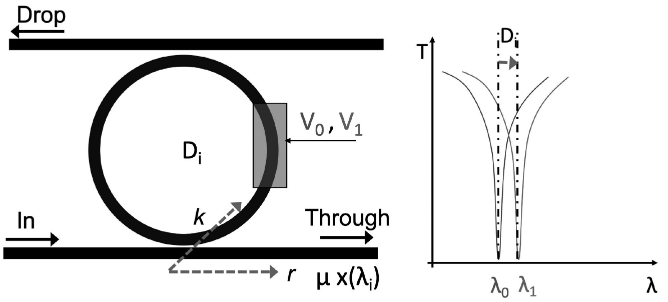

Fig. 3. Microring resonator (MRR) operation for performing point-wise multiplication.

using tunable optical resonators in a chip-based frequency comb generator [20]. Moreover, on the MRR side, our system can leverage on Dense-WDM (DWDM). This is achievable due to strong optical confinement of silicon waveguides using tunable MRRs with more than 50 nm Free Spectral Range (FSR) with Quality factors Q close to 10^4, which allow as many as 50 channels [21]. Assuming approximately 0.8 nm channel spacing, the resonance bandwidth can be broadened up to 0.4 nm, while maintaining an estimated cross-talk level of -10 dB [22].

Most of the modern neural networks have one or more fully-connected layers, which creates N^2 synaptic connections. On the other hand, 10 k connections are barely sufficient to even implement miniature fully-connected neural networks and simple benchmark datasets such as MNIST with 728 neurons only in the input layer. In contrast, CNNs benefit from sparse connections between the local input regions and filters. A common CNN architecture usually has filters of shape 3×3 up to 11×11 that connect receptive fields and the filters. From functional point of view smaller filters are favored over larger filters, as they are capable of detecting finer local patterns. Larger and global patterns are detected in the layers closer to the output of CNNs. These features are more abstract and are built on top of the previously low-level features.

We use the proposed scheme in Figure 2, to perform two heuristic Winograd transformations and one element-wise matrix multiplication (EWMM) on each wavelength. Figure 3 shows the details of a MRR weighting function that operates on a single wavelength λ_0. The MRR acts as a tunable analog filter centred at λ_0, in which the voltage applied to the EOM module lets only a portion of light to travel through the waveguide. The modulation can be triggered by an analog electric field generated by a memristor. In this work we use a memristor device which can store the weights with 6 bits of resolution [23]. The transmission spectrum (T) of the ring has a natural resonant frequency of λ_0. When WDM light passes through the coupled waveguide, the component with wavelength λ_0 is coupled into the ring. By raising the bias voltage to V$_1$, the resonant frequency shifts to λ_1 due to the change in the effective refractive index of the ring. The difference between V$_0$ and V$_1$ controls the difference between λ_0 and λ_1, i.e. the transmission (Δ_i). The variation of the transmission at λ_0 represents, in our scheme, the point-wise multiplication.

The most used MRR modulator has silicon based p-i-n junction that is side coupled to a wave guide as described in [24] or p-n junction reported [25]. Current silicon-based MRR modulators [26]–[28], as well as foundry level implementations, exhibit a speed up to 50 GHz, with a driving voltage of usually a few Volts (1–2 V) and an efficiency (Vπl) of few tenths of V cm. Experimental results that corroborate our estimation are reported in [29], where Silicon-based electro-optic MRRs exhibit a modulation in a working spectrum of 0.1 nm and a speed of 11 GHz while have an insertion loss of as low as only 2 dB. This by no means is a limiting factor in the inference stage, considering that the network has been trained and the weights are set. Therefore, the latency of the network is given by the time-of-flight of the photon.

Beside the uncertainty due to fabrication imperfections, which could be compensated, the main source of noise that affects a MRR modulator is electrical noise and, in this case, eventual non-ideality in setting the analog voltages with memristive device that could vary over time. Moreover, for high data rate situations(> 20 Gb/s), the intra-channel cross-talk becomes relevant, and power penalties need to be considered [30], [31]. Regarding the operating dynamic power, the maximum allowed optical power flowing in each physical channel of the photonic accelerator is bound by the optical power that would produce nonlinearities in the silicon waveguides and the minimum power that the photo-detector can distinguish from noise (SNR $= 1$). This sets the upper and lower operating range of photodiode, which we refer to as the dynamic range of the photodiode. Foundry level [32] integrated Germanium photo-diodes can reach up to 40 GHz with a responsivity of 0.6 A/W and a Noise equivalent power (NEP) of around 1pW/$\sqrt{\text{Hz}}$ operating in reverse bias (-2V). Research-level photodetectors working in the 100 s of GHz range have also been demonstrated [33]–[35]. However, the dynamic range of the photodiode needs to be accurately set to avoid saturation and account for the bit resolution [12]. For this scheme, according to the bit resolution, the estimated dynamic range is 20 dB.

The speed of the optical part of the accelerator, without considering the I/O interface, according to [36] is given by the total number of the MRR and their pitch. Photodetection and phase cross-talk are expected to be the main sources of error in the proposed scheme.

Another issue in using MRRs is attributed to variations in device fabrications, which can result in the spectral shift of the resonance frequency. The resonance frequency of MRRs can be tuned in multiple way. Due to high optical sensitivity of materials such as silicon to temperature, thermal tuning is the most widely adopted tuning technique for MRRs. This can be achieved by placing micro-heaters on top of each MRR [37], [38].

B. *Memristors as Analogue Weight Storage*

Neuromorphic systems inspired by human brain rely upon two major principles, namely massively distributed processing and proximity of local memory to these processing elements. These memory units demand some level of programability (plasticity), with their programming speed requirements being in the MHz regime. At this time, almost all state-of-the-art neural networks, perform the training and the inference phases separately. This means that once the weights are trained and set, for the inference

TABLE I
KERNEL SIZE BREAKDOWN IN STATE-OF-THE-ART CNNS. IT CAN BE SEEN THAT FILTER OF SIZE 5×5 COMPRISE ONLY A MINUTE FRACTION OF TOTAL FILTERS

CNN	1×1 (%)	3×3 (%)	Small 1D filters	5×5 (%)
GoogLeNet	64.9	17.5	1.7	15.9
Inception V3	43.2	17.9	35.7	3.2
Inception V4	40.9	16.1	43	0
MobileNet	93.3	6.7	0	0
ResNet50	68.5	29.6	1.9	0
VGG16	0	100	0	0

phase, one does not need to change the weights. In addition, weights in our proposed system are represented by an analog voltage bias of MRRs. A potential weight storage for our system should be analog and non-volatile with long retention time.

Having said that, memristive memory devices have attracted the attention of researchers due to their interesting characteristics including but not limited to non-volatility, long state retention time, and ultra-low power consumption [23], [39], [40]. Over the past few years the bit-resolution of such memristive memory devices has risen monotonically [41]–[44]. Recently, authors in [23] proposed, fabricated, and evaluated an analog multibit memristive memory with bit-resolution of up to 6.5 bits. Each memristive device takes up 20 $\mu \times 20$ μ in area and can retain the resistance state for up to 8 hours. In *AlexNet* the 3 rd convolutional layer has the largest number of convolutional filter weights equal to 884,736. Assuming overhead circuitry increases the footprint to approximately 50 $\mu \times 50$ μ, the memristive memory required for the largest layer of *AlexNet* can be realized in less than 0.25 cm^2.

IV. Fast Algorithms for Convolution Operation

As the name suggest the convolution operation account for the bulk of all operations in a CNN during both training and inference stages. However, each of the training and inference stages demand a different type of performance requirement. During the training, the emphasis is more on throughput rather than time. This is mainly due to the fact that the model under train needs to observe a large ”ensemble” of data, the batch, as fast as possible. Therefore, time is amortized over many inputs. On the other hand, during the inference stage, applications are mostly latency sensitive. For instance, in a self-driving car application only a few input image scenes are needed to be processed per second, but that is required to be at a very low latency timescale. Having said that, a neuromorphic processor designed for inference is expected to satisfy stringent timing requirements.

An important parameter that is shown to have a significant impact on the latency of CNNs is the size of their filters. It is generally known from a functional point of view that CNNs with smaller filters are preferred over CNNs with large filters [45]–[47]. Table I shows the breakdown of filter size for some of the state of the art CNN architectures. This is mainly due to the fact that small filters are better in finding local features without sacrificing the resolution. More abstract and more global features can be detected in higher layers of a CNN built on previous local layer features. As we discussed in section III, a physical implementation of photonic MRRs favors small size filters due to limited number of available wavelength bands. This synergy between functional and photonic realization of CNNs is the primary motivation behind this work.

At the time of writing this paper, there are three major ways to speed up the convolution operation. First, the General Matrix Multiplication (GEMM) approach, in which the convolution is converted to matrix multiplication operation using Toeplitz matrix. The downside to this method is that Toeplitz conversion expands the input by a factor of $r \times r$ where r is the size of the filter. Second method uses Fast Fourier Transform (FFT) to perform tiled convolution operations. From Fourier theorem we know that cyclic convolution can be performed by transforming the input and filters into Fourier domain. An element-wise multiplication (also known as Hadamard multiplication) results in an equivalent of convolution, but in Fourier domain. An inverse FFT operation transforms the calculated convolution back into the original domain. FFT-based convolution had been the method of choice for convolution operation [48]–[50] until the recent past. Lately, it is shown that FFT-based convolution is better suited for larger filter sizes [15]. The third method uses the Winograd filtering algorithm, which we explain in detail in the following section.

A. Winograd Algorithm

In a 2D convolution, a single output component of the convolution is calculated by,

$$y_{n,k,p,q} = \sum_{n=1}^{c}\sum_{x=1}^{r}\sum_{y=1}^{r} x_{n,c,p+x,q+y} \times w_{k,c,x,y} \tag{1}$$

The operation in equation 1 is repeated for all outputs convolution components. In a brute-force convolution the total number of multiplications required to perform a full convolution is equal to

$$(m \times r)^2 \tag{2}$$

where m is the size of the output feature map channel and r is the size of the filter. At the time of writing this paper, Winograd convolution in the most efficient convolution algorithm being used for CNNs [15]. Winograd convolution is based on the minimal filtering principles. The algorithm states that in order to calculate m outputs with a finite impulse response (FIR) filter of size r, denoted by $F(m, r)$, the number of required multiplications is,

$$n = (m + r - 1) \tag{3}$$

While equation 3 is derived for the 1D convolution operation, one can nest it with itself to acquire a 2D convolution. Therefore, the number of multiplications needed for the same 2D convolution is given by,

$$(m + r - 1)^2 \tag{4}$$

From equation 2 and 4 we can infer that Winograd results in a reduction in the complexity by a factor of,

$$\frac{(mr)^2}{(m+r-1)^2} \tag{5}$$

It should be noted that in our proposed photonic accelerator, multiplication operations are carried out by MRRs. Any reduction in the total number of multiplication operations, and thus MRRs, can save us not only in footprint of the design, but also in the design complexity.

Now, in order to understand how minimal Winograd works, let us first consider the case for 1D convolution. Let matrix W be the matrix of weights, and matrix D be the data matrix. Winograd computes the $F(2,1)$ convolution as following

$$\begin{bmatrix} d_0 & d_1 & d_2 \\ d_1 & d_2 & d_3 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} m_1 + m_2 + m_3 \\ m_2 - m_3 - m_4 t \end{bmatrix} \tag{6}$$

where values m_i are intermediate values found by

$$m_1 = (d_0 - d_2) \times w_0$$

$$m_2 = (d_1 + d_2) \times \frac{w_0 + w_1 + w_2}{2}$$

$$m_3 = (d_2 - d_1) \times \frac{w_0 - w_1 + w_2}{2}$$

$$m_4 = (d_0 - d_3) \times w_2$$

Above equations show that with only 4 multiplications between inputs and weights, Winograd can compute a $F(2,3)$ convolution. All w_i terms can be pre-computed after the training stage. In order word, during the inference time, while data values d_i, corresponding to inputs change, the w_i values remain the same throughout the inference. The 1D Winograd can be expressed by a closed matrix form as

$$Y = A^T[(G \times w) \odot (B^T \times d)] \tag{7}$$

where A^T, B^T, and G are three heuristic transforms described by equations 8, 9, and 10. w is the weight vector and d is the input vector.

$$A^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & -1 \end{bmatrix} \tag{8}$$

$$B^T = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix} \tag{9}$$

$$G = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

One conclusion from equation 6 is that to compute a single output of 1D convolution only a window of $(m+r-1)$ input values are needed.

In a modern CNN the bulk of convolution operations are comprised of 2D convolutions. Equation 7 can be easily extrapolated for 2D convolution by nesting two 1D Winograd convolutions. The resulting 2D Winograd would be,

$$Y = A^T[(G \times w \times G^T) \odot (B^T \times d) \times B] \tag{11}$$

From [15] for the case of $F(4 \times 4, 3 \times 3)$ matrices B^T, G, and A^T have the forms,

$$A^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & 2 & -2 & 0 \\ 0 & 1 & 1 & 4 & 4 & 0 \\ 0 & 1 & -1 & 8 & -8 & 0 \end{bmatrix} \tag{12}$$

$$B^T = \begin{bmatrix} 4 & 0 & -5 & 0 & 1 & 0 \\ 0 & -4 & -4 & 1 & 1 & 0 \\ 0 & 4 & -4 & -1 & 1 & 0 \\ 0 & -2 & -1 & 2 & 1 & 0 \\ 0 & 4 & 0 & -5 & 0 & 1 \end{bmatrix} \tag{13}$$

$$G = \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ \frac{-1}{6} & \frac{-1}{6} & \frac{-1}{6} \\ \frac{-1}{6} & \frac{1}{6} & \frac{-1}{6} \\ \frac{1}{24} & \frac{1}{12} & \frac{1}{6} \\ \frac{1}{24} & \frac{-1}{12} & \frac{1}{6} \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

The number of addition and multiplications required for Winograd transform, not the element-wise multiplications, increases quadratically with the tile size. Thus, Winograd is expected to perform most efficiently for smaller filter sizes, and thus smaller input tiles.

V. Architecture Design

In this paper we propose a photonic CNN accelerator based on Winograd algorithm and realized using the photonic neuron introduced in [19]. Figure 4 depicts the architecture of a single Winograd PE. Our proposed accelerator processes a single layer of a CNN at a time. This is mainly due to the fact that in a CNN different tiles of output feature maps are computed sequentially, and thus arrive at different times. But, in order to initiate processing of the next layer, all the inputs from the previous layer need to be available and synchronized. Our approach to process one layer at a time enforces this synchronization. Furthermore, implementing multiple layers of a CNN will result in large area overheads.

At the input of our accelerator, an input tile of shape $n \times n \times c$ along with filters of size $r \times r \times c$ are transformed into the Winograd domain. Input and filters' transforms are then multiplied element by element. The output of this multiplication needs to be transformed back using an inverse Winograd transform. The signals at this stage are digitized using an array of ADCs and placed onto the output line buffers to be stored back in the off-chip memory.

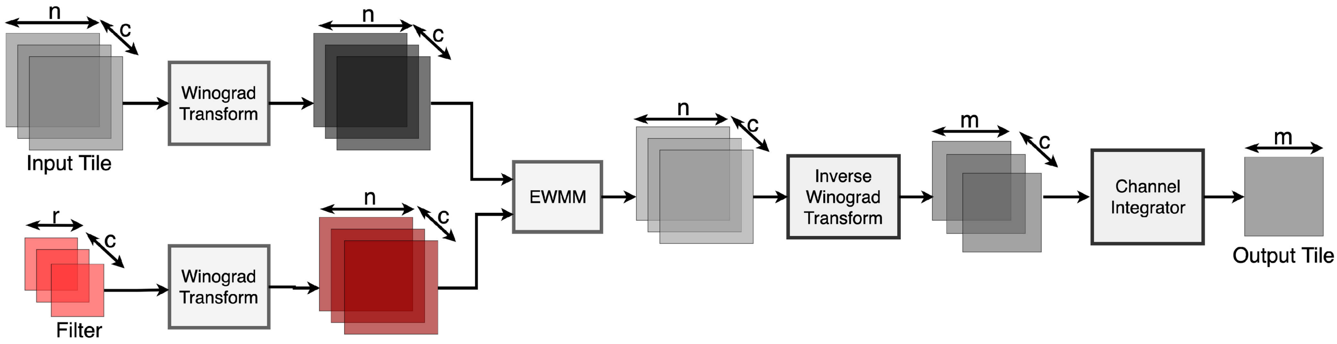

Fig. 4. High-level flow diagram of Winograd filtering technique for convolution operation. Unlike conventional convolution, which computes a single output at a time, Winograd algorithm computes a tile of output, here of size $m \times m$. In order to generate an output tile, Winograd requires to fetch an input tile of size $n \times n$. Both input tile and filter are transformed into the Winograd domain. Within the Winograd domain, previously transformed input and filter are multiplied in an element-wise fashion. Finally, the output of the element-wise multiplication is transformed back into the original domain and channels are collapsed into a single value per tile element.

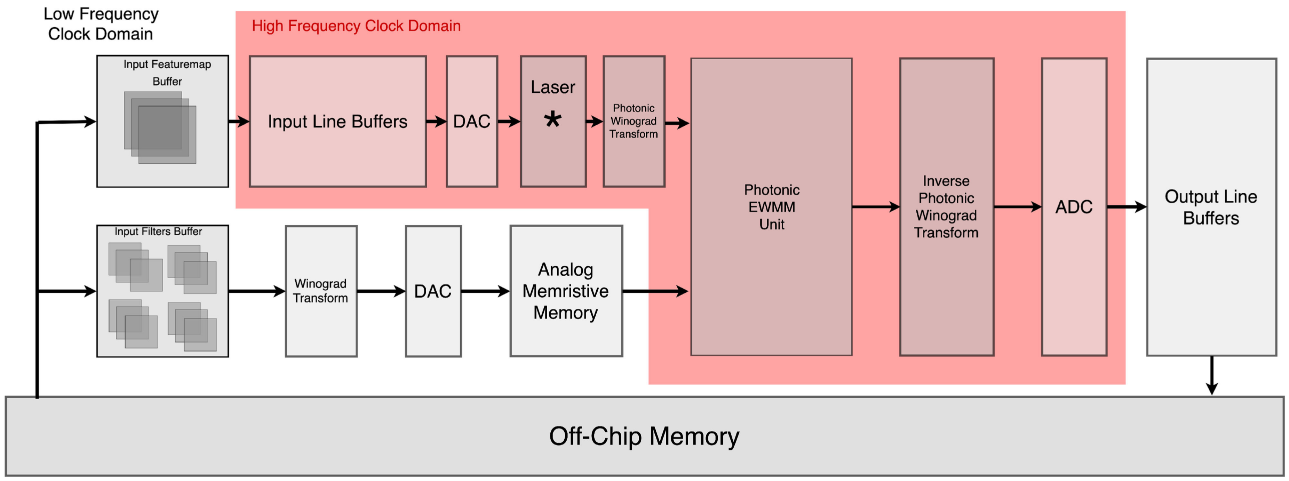

Fig. 5. High-level architecture of our proposed photonic accelerator. Input feature-maps and filters are initially stored in an off-chip memory. Input tiles of size $n \times n$ are loaded into the input line buffer one at a time. Kernel weights do not change once the CNN is trained. Thus, we perform filters' Winograd in electronics and the cost is amortized over many input tiles. Winograd transform for input feature map tiles are computed in photonics. The photonic element-wise matrix multiplication (EWMM) unit performs the core Winograd element-wise Winograd multiplications. Outputs are digitized and placed onto the output line buffer. Finally, processed layer outputs are stored in the off-chip memory.

Figure 5 presents the overview of the our proposed architecture. Our proposed architecture runs on two clock domains. First a high speed 5 GHz clock domain, which accommodates low latency components of the accelerator including the photonic components. In section VI-A we explain our rationale on how we arrive at the 5 GHz high speed clock frequency. The rest of the accelerator including input feature map buffers, filter buffers, filter Winograd DSP module, and filter path DAC run on a slower clock domain because there is no time sensitivity on filter path, and data transfer from/to off-chip memory. At the heart of our accelerator, we have an Element-Wise Matrix Multiplication unit, which we implement in photonics using photonic neurons. We store the input feature maps and filters in an off-chip memory. Both the input feature maps and filters require to go through Winograd transformation, which are matrix multiplications described in equations 13 and 14. It should be noted that while input feature maps change for different tiles of inputs, filters are fixed for each layer. For that, we implemented the input feature map transformations in photonics and filter Winograd transformations in electronic DSP. This way, we will not pay the overhead associated with photonic implementations including the conversion of electronic filters to photonics. Later, the transformed filters and input feature map tiles are converted into analog signals to modulate the laser beams. However, as the filters are fixed over the processing time of the layer, analog filter signals need to be maintained for that time. Thus, we propose to use the non-volatile analog memristive memory bank, which maintains these voltages in their analog form for long retention times.

In Winograd convolution, and in each iteration, a tile of $n \times n$ is processed. In order to process an entire feature map, the transformed filter tile needs to move across the input feature map. In this paradigm two successive input tiles share size $(r - 1) \times n$ elements. This, introduces data reuse opportunities, to

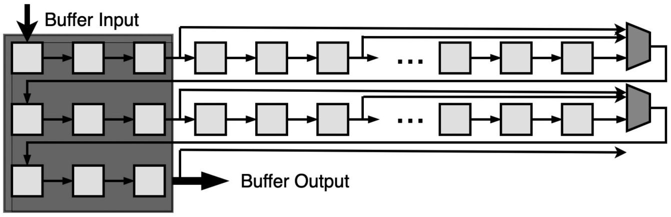

Fig. 6. An example line buffer design to load and hold an input tile of size 3×3.

avoid multiple queries of same data block. Here our goal is to exploit this opportunity at the front-end of our accelerator. Our design is inspired by the work in [16], where authors utilize line buffers to avoid redundant queries from the off-chip memory. Figure 6 shows an example line buffer design to load and hold a 3×3 input feature map tile. Input tiles are fetched from off-chip memory and loaded into the line buffer. Buffered tiles are then passed into the digital to analog converter (DAC) using parallel channels.

In parallel to the input data stream, transformed filter weights are converted into the analog signals to program the analog memristive memory. We then use the voltage generated using the stored analog signals to modulate the laser source for the filters. Each output signal generated by a DAC is then used to modulate a laser beam of a particular wavelength λ_i. It is worth noting that for each set of filters modulated by the laser source, input line goes through multiple iterations corresponding to different input tiles. Once both input tile laser beam and the filter laser beam are ready, the EWMM, multiplies each element of the Winograd input feature map tile with its corresponding Winograd filter value. The output from EWMM unit must be transformed back from Winograd domain into the original domain by the inverse Winograd transform. The result contains output feature map tiles for multiple channels c. Lastly, output feature map values are digitized and stored back into the off-chip memory.

A key principle in HPC is to try to minimize the IO and other communication latencies, compared to that of the computation time, to avoid unit under-utilization. From Algorithm 1 we can see that the two inner-most loops iterate over different channels of the input feature map tile and different filters. Moreover, operations within these two loops are independent form one another. This provides parallelization opportunities at the cost of additional hardware. In other words, the amount of parallelization and speed up we can achieve, scales linearly with the number of pipeline replications in our system. This linear scaling plateaus as soon as the computation bandwidth approaches the data transfer bandwidth. Our envisioned design uses an arbitrary number of 100 parallel paths. Our evaluation results in the next section justifies this selection.

VI. Evaluation

In this section we evaluate the performance of our accelerator for the 3×3 filters of the *VGG16* network against the recent FPGA [16], [51]–[54] and GPU implementations [16].

Algorithm 1: Winograd for 2D convolution.

```
for row=0; row<H; row+=m do
    for column=0; column<H; column+=m do
        for channel=0; channel<c; channel+=1 do
            for filter=0; filter<N; kerne;+=1 do
                load input tile;
                transform input tile;
                load transformed filter;
                perform EWMM
            end
        end
    end
    Output Winograd convolution result
end
```

A. Speed

Here we develop a model to estimate the execution time of the our accelerator. First we model the time required to convolve one input tile with one filter and we call it T_{tile_filter}. Following that, we generalize the model to the case where we parallelize the process based on available resources. For one input feature map tile and a filter, both, the input branch and the filter branch of the Figure 5 are fully pipelined. Therefore, the execution time of a layer is determined by the longer of the two paths of filter path and input path. For each iteration of the filter path, input data path goes through multiple iterations. This is because a single filter operates on many input data tiles. That said, the input data path sets the upper bound on the delay. Our execution time model is comprised of two major components namely, the Input/Output time (T_{IO}) and the computation time (T_{Comp}). We define (T_{IO}) as.

$$T_{IO} = max(T_{load}, T_{offload}) \tag{15}$$

where T_{load} is the time it takes to transfer data from the off-chip memory to the input of the laser sources. Moreover, we can implement a total of P_{input} DACs to speedup the data transfer. Considering the fact that our input matrices are of the shape 4×4, we used an array of 16 DACs in this work. Similarly $T_{offload}$ is the time to store back the computed outputs from the inverse Winograd transform to the off-chip memory. The goal is to match the rate of the ADC at the output with input DAC to avoid any speed mismatch and thus congestion in the pipeline. At the time of this review both on-chip DACs and ADCs are capable of operating at sampling rates of more than 18 GS/s for bit resolution of at least 8 bits [24], [55]. Furthermore, with recent advances in memory technology, current memories are able to transfer data at high IO bandwidths up to more than 512 Gb/s [56]. This high memory bandwidth allows us to buffer data and filters from off-chip memory at high transfer rates and feed it to our input line buffers. However, for our line buffers we need memories with high clock frequency and short access time. Current reported memory technologies have access time as short as 200 ps.

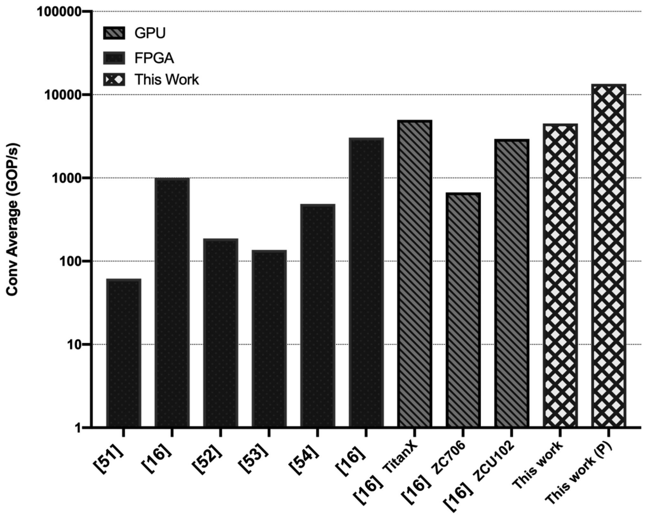

Fig. 7. Comparison of convolution operation speed for FPGA, GPU, and our photonic implementation. The last column labeled with (p) represents the speed of the photonic core in the absence of electronics.

At the photonic core of our accelerator, $T_{compute}$ is,

$$T_{compute} = T_{laser} + T_{Winograd} + T_{EWMM} + T_{iWinograd} \tag{16}$$

where $T_{Winograd}$ is the time to compute the Winograd transform, T_{EWMM} is the time to perform the element-wise matrix multiplication, and $T_{iWinograd}$ is the time compute the inverse Winograd transform. Once the laser is set up, input signals only incur a time delay equivalent to flight time of the light before they are fed into the ADC. Having said that the clock frequency of the pipeline is determined by

$$frequency_{clock} \leq \frac{1}{min(T_{load}, T_{offload}, T_{compute})} \tag{17}$$

As a result of equation 17, we picked a clock frequency of 5 Ghz, which satisfies equation 17. From equation 17 T_{tile_filter} is simply found by,

$$T_{tile\_filter} = \frac{1}{5\text{ GHz}} = 200\ ps \tag{18}$$

For a $F(4 \times 4, 3 \times 3)$ Winograd, each T_{tile_filter} returns an output block of size 9 equivalent to 9 convolution operations. Having a clock frequency of 5 GHz, our proposed accelerator performs at 9 × 5G = 45 GOP/s. Figure 7 shows the average convolution speed comparison of our proposed accelerator against the state-of-the-art FPGA and GPU implementations.

B. Power

In order to estimate the dynamic power consumption of our proposed system, we built our in-house estimator by augmenting the standard Google Tensorflow tool. While primarily used for training and inference stages of neural networks, at the core, Tensorflow is a symbolic mathematical graph processing platform. Tensorflow enables users to express arbitrary computations into a dataflow graph, which is extremely useful in the context of neural networks. However, out-of-the-box Tensorflow is completely agnostic to physical realization of the neural networks being implemented. Thus, we augmented Tensorflow high-level API with mathematical models of electro-optical components. Figure 8 depicts the native Tensorflow toolkit hierarchy against our augmented version. In our estimator, each primitive mathematical operation is given two physical models namely, the power model and the noise model. While, the noise model can impact the functionality, thus accuracy, in a neural network, the power model only models/measures consumed power. Table II shows some of these mathematical operations mapped to to their physical realizations.

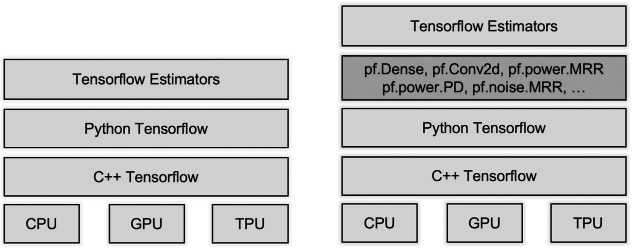

Fig. 8. High level Tensorflow toolkit hierarchy vs. augmented Tensorflow.

TABLE II
MAPPING OF PRIMITIVE MATH OPERATIONS TO THEIR HARDWARE REALIZATION

Math Operation	Photonic Representation
Addition	Photodiode
Multiplication	MRR
Connection	Waveguide
Non-linear Activation	Electro-absorption Modulator

Photodiode power can be simply derived from its Responsivity equation:

$$R = \frac{I_{ph}}{P_{in}} = \lambda \frac{q}{hc} \eta \quad \left[\frac{A}{W}\right] \tag{19}$$

where I_{ph} is the photocurrent, P_{in} is the optical input signal power, q is the electron charge, λ is the wavelength, h is the Planck's constant, and c is the speed of light. It should be noted that the signal is encoded in the optical input power P_{in}. In this work use Aim photonics PDK values in [57]. Similarly, we modeled both thermal noise and the shot noise in photodiode.

$$I_{sn} = \sqrt{2q(I_{ph} + I_D)\Delta f} \tag{20}$$

$$I_{tn} = \sqrt{\frac{4K_B T \Delta f}{R_{SH}}} \tag{21}$$

where I_D is the dark current of the photodetector, Δf is the noise measurement bandwidth, K_B is the Boltzmann Constant, T is temperature in Kelvins and R_{SH} is total equivalent shunt resistance of the photodiode. For MRRs we accounted for per unit length propagation loss.

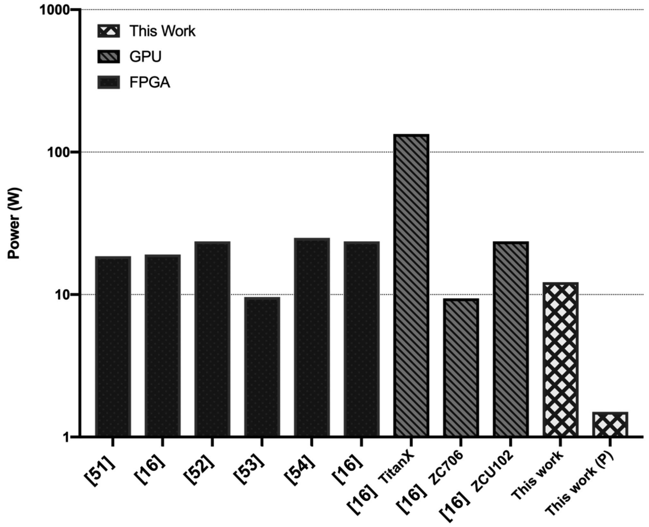

Fig. 9. Comparison of convolution operation power for FPGA, GPU, and our photonic implementation. The last column labeled with (p) represents the power consumption of the photonic core in the absence of electronics.

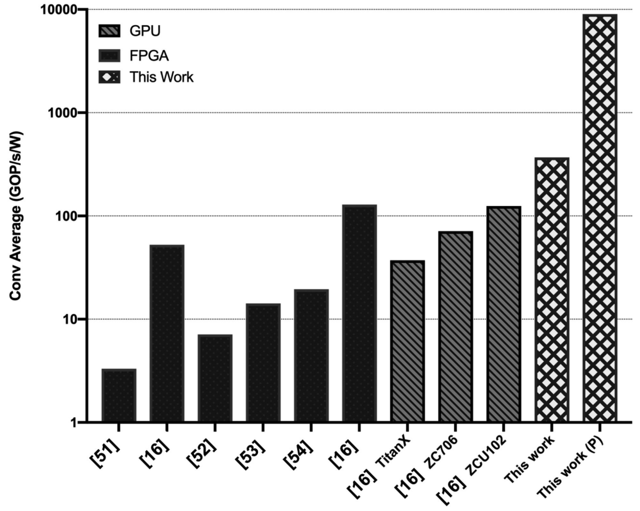

Fig. 10. Comparison of energy efficiency for FPGA, GPU, and our photonic implementation. The last column labeled with (p) represents the power consumption of the photonic core in the absence of electronics. The results show that using photonics as an accelerator has the potential of improving energy efficiency by up to more than three orders of magnitude.

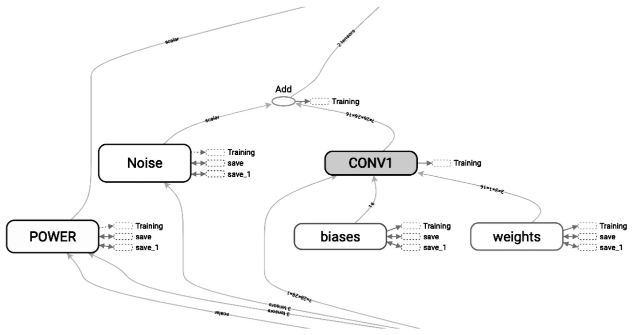

Fig. 11. Visualalization of an augmented convolutional layer using power and noise models for *VGG16* network.

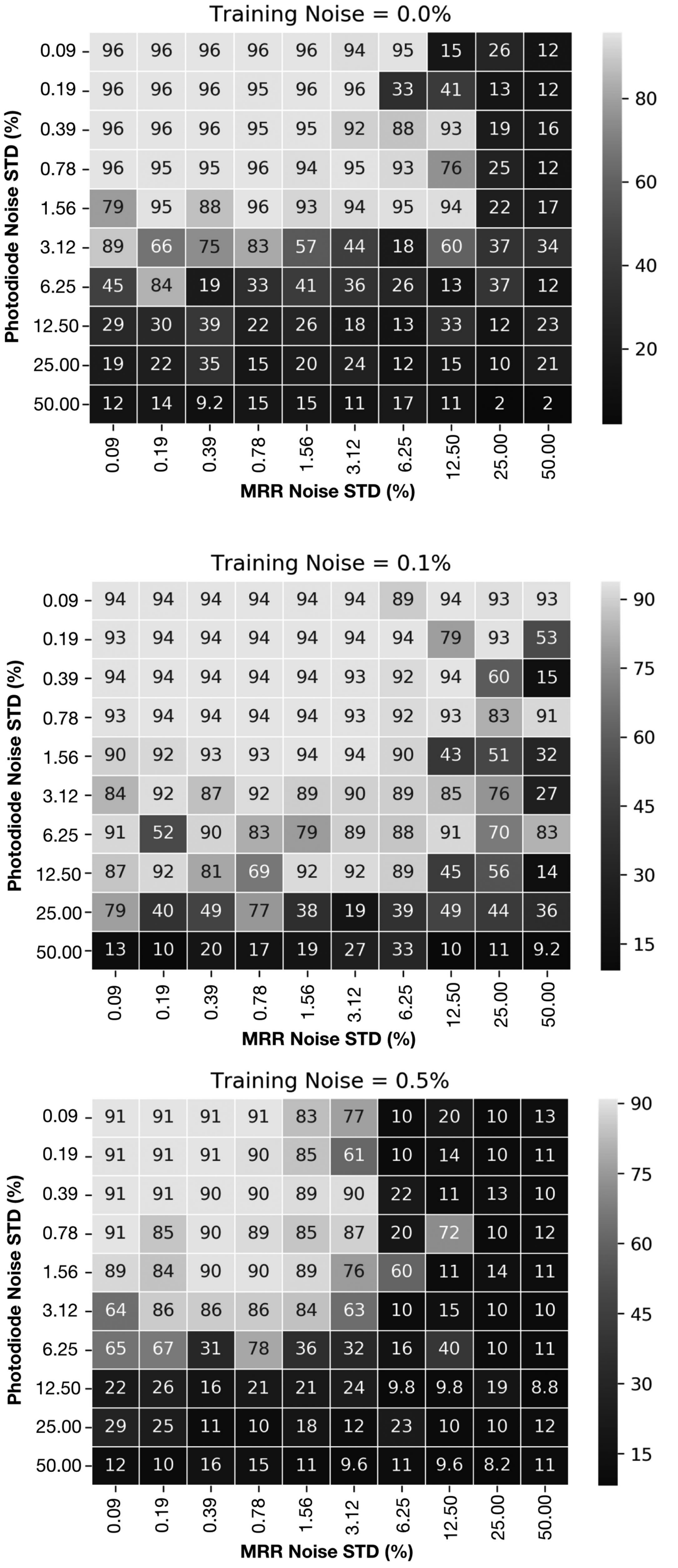

Fig. 12. The evaluation of the effect of physical photodiode and MRR noise on inference accuracy. This effect can be partially compensated through introduction of an artificial noise source during the training stage. At the absence of training noise source (top) inference accuracy is quickly deteriorated as we sweep the photodiode and MRR noise. By introducing an equivalent of 0.1% guassian noise, the network becomes more robust to inference noise. Further increase in training noise level (bottom) hinders the network from proper training.

Figure 9 depicts the power comparison results. Finally We plotted the energy efficiency figure of merit defined by the ratio of speed to power in Figure 10.

VII. Training, Inference, and Noise

We initially trained our neural network offline on a conventional digital computer. Later during the inference stage we loaded the trained weights into our in-house simulator, which is equipped with noise sources modeling. Our hypothesis was that inference on a noisy neural network would result in some loss in accuracy. This is mostly due to the fact that, the network used during the training is noise-less, with 32-bit floating point resolution, while during inference the weights all in a sudden face a noisy network. In other words, the network performing inference experiences unseen noise behavior that results in accuracy loss. We tested our hypothesis by sweeping a range of inference noise levels and observing its effect on accuracy. For that reason, we identified two major noise sources, namely the neuron output noise and the weight noise. The neuron output noise represents the noise introduced at the output of each neuron by the photodiode and the nonlinear activation function. The first plot in Figure 12 shows how accuracy is impacted by noise during inference for the case that the network was trained free of any noise source.

Our next hypothesis was that, if we allow for certain amount of noise during the training, the model would become more robust to noise during the inference stage. To that end, we trained the network with output noise source on. We only added the output noise, and left the weight noise off, because weights are required to be calculated with maximum precision during training. In fact we observed that even a minute amount of noise added to the weights during the training could destroy the accuracy of the network to its baseline level of about 10%. We swept the addition of training noise at logarithmic steps from 0.01% to 1%. Figure 12 depicts the effect of adding an output noise equivalent to 0.1% and 0.5% of the maximum signal swing at the output of neurons. In our experiments we observed that the addition of about 0.1% noise during the training may result in slight 2% accuracy loss for low level of noise during inference. However, the model becomes more robust to higher levels of inference noise. This shows that modeling noise by addition of noise during training can fine-tune the network for a physical noisy realization as shown in Figure 12 (middle). Lastly, we noticed adding further amount of training noise beyond the initial 0.1% resulted very significant inference accuracy losses shown in Figure 12 (bottom).

VIII. Conclusion

In this paper we presented a photonic CNN accelerator based on Winograd filering convolution algorithm. Winograd reduces the total number of multiplication, thus hardware, to perform convolution operation. We evaluated the speed of our accelerator by developing an analyical framework. Our results show that a photonic accelerator can compete with state-of-the-art Winograd based FPGA and GPU implementations. Such photonic accelerator has the potential of improving the energy efficiency by up to three orders of magnitude. However, the overall speed is bound by the limitations of IO and conversions in DAC and ADC. To evaluate power performance we augmented the native hardware-agnostic Google Tensorflow tool with power models of our hardware components. Similar to speed performance, electronic IO and convertors are the major consumers of power in our proposed design. However, the photonic core, without the electronic interface, can operate while consuming up to two orders of magnitude less power. In addition, we modeled noise into our Tensorflow-based simulator, to investigate the effect of hardware noise sources such as photodiode noise and MRR noise on the functionality (accuracy) of our CNN. We found training the CNN with a small noise component, 0.1% of the signal swing in our experiment, can result in the CNN become more robust to inference-time noise introduced by noisy photodiodes and MRRs.

References

[1] P. R. Prucnal and B. J. Shastri, *Neuromorphic Photonics*. Boca Raton, FL, USA: CRC Press, May 2017.

[2] I. Chakraborty, G. Saha, A. Sengupta, and K. Roy, "Toward fast neural computing using all-photonic phase change spiking neurons," *Sci. Rep.*, vol. 8, no. 1, 2018, Art. no. 12980.

[3] J. Feldmann, N. Youngblood, C. Wright, H. Bhaskaran, and W. Pernice, "All-optical spiking neurosynaptic networks with self-learning capabilities," *Nature*, vol. 569, no. 7755, pp. 208–214, 2019.

[4] J. K. George *et al.*, "Neuromorphic photonics with electro-absorption modulators," *Opt. Express*, vol. 27, no. 4, pp. 5181–5191, Feb. 2019.

[5] C. Wang *et al.*, "Integrated lithium niobate electro-optic modulators operating at CMOS-compatible voltages," *Nature*, vol. 562, no. 7725, pp. 101–104, Oct. 2018.

[6] A. Liu *et al.*, "A high-speed silicon optical modulator based on a metaloxidesemiconductor capacitor," *Nature*, vol. 427, no. 6975, pp. 615–618, Feb. 2004.

[7] R. Amin *et al.*, "0.52 V mm ITO-based Mach–Zehnder modulator in silicon photonics," *APL Photon.*, vol. 3, no. 12, Dec. 2018, Art. no. 126104.

[8] H. Bagherian *et al.*, "On-chip optical convolutional neural networks," 2018, *arXiv:1808.03303*.

[9] A. Mehrabian, Y. Al-Kabani, V. J. Sorger, and T. El-Ghazawi, "Pcnna: A photonic convolutional neural network accelerator," in *Proc. 31st IEEE Int. Syst.-Chip Conf.*, 2018, pp. 169–173.

[10] W. Liu *et al.*, "Holylight: A nanophotonic accelerator for deep learning in data centers," in *Proc. IEEE Des., Autom., Test Eur. Conf. Exhib.*, 2019, pp. 1483–1488.

[11] "Optalysys." [Online]. Available: https://www.optalysys.com/

[12] Y. Shen *et al.*, "Deep learning with coherent nanophotonic circuits," *Nature Photon.*, vol. 11, no. 7, pp. 441–446, Jul. 2017.

[13] T. W. Hughes, M. Minkov, Y. Shi, and S. Fan, "Training of photonic neural networks through in situ backpropagation and gradient measurement," *Optica*, vol. 5, no. 7, pp. 864–871, Jul. 2018.

[14] M. Miscuglio *et al.*, "All-optical nonlinear activation function for photonic neural networks [Invited]," *Opt. Mater. Express*, vol. 8, no. 12, pp. 3851–3863, Dec. 2018.

[15] A. Lavin and S. Gray, "Fast algorithms for convolutional neural networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 4013–4021.

[16] L. Lu, Y. Liang, Q. Xiao, and S. Yan, "Evaluating fast algorithms for convolutional neural networks on FPGAS," in *Proc. IEEE 25th Annu. Int. Symp. Field-Programmable Custom Comput. Mach.*, 2017, pp. 101–108.

[17] Y. Shen *et al.*, "Deep learning with coherent nanophotonic circuits," *Nature Photon.*, vol. 11, no. 7, pp. 441–446, 2017.

[18] A. N. Tait, M. A. Nahmias, B. J. Shastri, and P. R. Prucnal, "Broadcast and weight: An integrated network for scalable photonic spike processing," *J. Lightw. Technol.*, vol. 32, no. 21, pp. 4029–4041, Nov. 2014.

[19] A. N. Tait *et al.*, "Microring weight banks," *IEEE J. Sel. Topics Quantum Electron.*, vol. 22, no. 6, pp. 312–325, Nov./Dec. 2016.

[20] H. Hu *et al.*, "Single-source chip-based frequency comb enabling extreme parallel data transmission," *Nature Photon.*, vol. 12, no. 8, pp. 469–473, 2018.

[21] Q. Xu, D. Fattal, and R. G. Beausoleil, "Silicon microring resonators with 1.5-μm radius," *Opt. Express*, vol. 16, no. 6, pp. 4309–4315, 2008.

[22] L. Chen, K. Preston, S. Manipatruni, and M. Lipson, "Integrated GHz silicon photonic interconnect with micrometer-scale modulators and detectors," *Opt. Express*, vol. 17, no. 17, pp. 15 248–15 256, 2009.

[23] S. Stathopoulos *et al.*, "Multibit memory operation of metal-oxide bi-layer memristors," *Sci. Rep.*, vol. 7, no. 1, 2017, Art. no. 17532.

[24] B. Xu, Y. Zhou, and Y. Chiu, "A 23-mw 24-gs/s 6-bit voltage-time hybrid time-interleaved ADC in 28-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 52, no. 4, pp. 1091–1100, Apr. 2017.

[25] M. Ziebell *et al.*, "Ten Gbit/s ring resonator silicon modulator based on interdigitated PN junctions," *Opt. Express*, vol. 19, no. 15, pp. 14 690–14 695, Jul. 2011.

[26] F. Y. Gardes *et al.*, "High-speed modulation of a compact silicon ring resonator based on a reverse-biased PN diode," *Opt. Express*, vol. 17, no. 24, pp. 21 986–21 991, Nov. 2009.

[27] "OSA | Ten Gbit/s ring resonator silicon modulator based on interdigitated PN junctions." [Online]. Available: https://www.osapublishing.org/oe/abstract.cfm?uri=oe-19-15-14690

[28] T. Baba *et al.*, "50-Gb/s ring-resonator-based silicon modulator," *Opt. Express*, vol. 21, no. 10, pp. 11 869–11 876, May 2013.

[29] P. Dong *et al.*, "Low V_pp, ultralow-energy, compact, high-speed silicon electro-optic modulator," *Opt. Express*, vol. 17, no. 25, Dec. 2009, Art. no. 22484.

[30] H. Jayatilleka *et al.*, "Crosstalk in SOI microring resonator-based filters," *J. Lightw. Technol.*, vol. 34, no. 12, pp. 2886–2896, Jun. 2016.

[31] M. Bahadori *et al.*, "Crosstalk penalty in microring-based silicon photonic interconnect systems," *J. Lightw. Technol.*, vol. 34, no. 17, pp. 4043–4052, Sep. 2016.

[32] "imec-ePIXfab SiPhotonics Passives." [Online]. Available: http://www.europractice-ic.com/SiPhotonics_technology_IHP_passives.p%hp

[33] L. Vivien *et al.*, "Zero-bias 40gbit/s germanium waveguide photodetector on silicon," *Opt. Express*, vol. 20, no. 2, pp. 1096–1101, Jan. 2012.

[34] Y. Salamin *et al.*, "100 GHz plasmonic photodetector," *ACS Photon.*, vol. 5, no. 8, pp. 3291–3297, Aug. 2018.

[35] P. Ma *et al.*, "100 GHz photoconductive plasmonic germanium detector," in *Proc. Int. Conf. Lasers Electro-Opt.*, May 2018, Paper SM2I.3.

[36] A. N. Tait *et al.*, "Neuromorphic photonic networks using silicon photonic weight banks," *Sci. Rep.*, vol. 7, no. 1, Aug. 2017, Art. no. 7430.

[37] W. Bogaerts *et al.*, "Silicon microring resonators," *Laser Photon. Rev.*, vol. 6, no. 1, pp. 47–73, 2012.

[38] X. Xue *et al.*, "Thermal tuning of kerr frequency combs in silicon nitride microring resonators," *Opt. Express*, vol. 24, no. 1, pp. 687–698, 2016.

[39] C. Yoshida, K. Tsunoda, H. Noshiro, and Y. Sugiyama, "High speed resistive switching in Pt/TiO$_2$/TiN film for nonvolatile memory application," *Appl. Phys. Lett.*, vol. 91, no. 22, 2007, Art. no. 223510.

[40] J. Borghetti *et al.*, "Memristiveswitches enable statefullogic operations via material implication," *Nature*, vol. 464, no. 7290, pp. 873–876, 2010.

[41] I. Baek *et al.*, "Highly scalable nonvolatile resistive memory using simple binary oxide driven by asymmetric unipolar voltage pulses," in *Proc. IEEE Int. Electron Devices Meeting*, 2004, pp. 587–590.

[42] E. J. Merced-Grafals, N. Dávila, N. Ge, R. S. Williams, and J. P. Strachan, "Repeatable, accurate, and high speed multi-level programming of memristor 1T1R arrays for power efficient analog computing applications," *Nanotechnology*, vol. 27, no. 36, 2016, Art. no. 365202.

[43] A. Prakash, D. Deleruyelle, J. Song, M. Bocquet, and H. Hwang, "Resistance controllability and variability improvement in a taox-based resistive memory for multilevel storage application," *Appl. Phys. Lett.*, vol. 106, no. 23, 2015, Art. no. 233104.

[44] S. R. Lee *et al.*, "Multi-level switching of triple-layered TaOx RRAM with excellent reliability for storage class memory," in *Proc. Symp. VLSI Technol.*, 2012, pp. 71–72.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 770–778.

[47] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2818–2826.

[48] M. Mathieu, M. Henaff, and Y. LeCun, "Fast training of convolutional networks through FFTs," 2013, *arXiv:1312.5851*.

[49] N. Vasilache *et al.*, "Fast convolutional nets with FBFFT: A GPU performance evaluation," 2014, *arXiv:1412.7580*.

[50] S. Chetlur *et al.*, "CUDNN: Efficient primitives for deep learning," 2014, *arXiv:1410.0759*.

[51] C. Zhang *et al.*, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2015, pp. 161–170.

[52] J. Qiu *et al.*, "Going deeper with embedded FPGA platform for convolutional neural network," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2016, pp. 26–35.

[53] N. Suda *et al.*, "Throughput-optimized opencl-based FPGA accelerator for large-scale convolutional neural networks," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2016, pp. 16–25.

[54] C. Zhang *et al.*, "Caffeine: Towards uniformed representation and acceleration for deep convolutional neural networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 38, no. 11, pp. 2072–2085, Nov. 2019.

[55] A. Nazemi *et al.*, "3.4 a 36gb/s PAM4 transmitter using an 8b 18gs/s DAC in 28nm CMOS," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2015, pp. 1–3.

[56] D. U. Lee *et al.*, "A 1.2 v 8 gb 8-channel 128 gb/s high-bandwidth memory (HBM) stacked dram with effective I/O test circuits," *IEEE J. Solid-State Circuits*, vol. 50, no. 1, pp. 191–203, Jan. 2015.

[57] "Silicon photonics process design kit (apsuny pdkv3.0)." [Online]. Available: http://www.aimphotonics.com/pdk

Armin Mehrabian received the B.S. degree in electrical engineering from Shahid Beheshti University, Tehran, Iran, focusing on Analog Electronics, and the M.S. degree in computer engineering from the George Washington University (GWU), Washington, DC, USA, focusing on VLSI and digital electronics design. He is currently working toward the Ph.D. degree in electrical engineering with the GWU, Washington, DC, USA. His research interests include high performance computing (HPC), neuromorphic computing, artificial intelligence from both software and hardware point of view. His current research involves leveraging nanophotonics for HPC architecture designs.

Mario Miscuglio received the master's degree in electric and computer engineering from the Polytechnic of Turin, Turin, Italy, and the Ph.D. degree in optoelectronics from the University of Genova (IIT), Genoa, Italy. He is a Postdoctoral Researcher with the Electrical Engineering Department, George Washington University, Washington, DC, USA. He is working as a Researcher with Harvard/MIT and also working as a Research Fellow with the Molecular Foundry, LBNL. His research interests extend across science and engineering, including photonic neuromorphic computing, nano-optics and plasmonics.

Yousra Alkabani received the B.Sc. and M.Sc. degrees in computer and systems engineering from Ain Shams University, Cairo, Egypt, in 2003 and 2006, respectively, and the Ph.D. degree in computer science from Rice University, Houston, TX, USA, in December 2010. She has been an Assistant Professor of computer and systems engineering with Ain Shams University since May 2011 and a Visiting Assistant Professor of computer science and engineering with the American University in Cairo, since 2013. Her research interests include hardware security, low power design, and embedded systems.

Volker J. Sorger received the Ph.D. degree from the University of California Berkeley, Berkeley, CA, USA. He is an Associate Professor with the Department of Electrical and Computer Engineering and the Leader of the Orthogonal Physics Enabled Nanophotonics (OPEN) Lab with the George Washington University, Washington, DC, USA. His research areas include opto-electronic devices, plasmonics and nanophotonics and photonic analog information processing and neuromorphic computing. Among his breakthroughs are the first demonstration of a semiconductor plasmon laser, attojoule-efficient modulators, and PMAC/s-fast photonic neural networks and near real-time analog signal processors. Dr. Sorger has received multiple awards among are the Presidential Early Career Award for Scientists and Engineers (PECASE), the AFOSR Young Investigator Award (YIP), the Hegarty Innovation Prize, and the National Academy of Sciences Award of the year. He is the Editor-in-Chief of Nanophotonics, the OSA Division Chair for Photonics and Opto-Electronics and serves at the board-of-meetings at OSA & SPIE, and the scholarship committee. He is a Senior Member of OSA and SPIE.

Tarek El-Ghazawi is a Professor with the Department of Electrical and Computer Engineering, George Washington University, Washington, DC, USA, where he leads the university-wide Strategic Program in High-Performance Computing. He is the Founding Director of The GW Institute for Massively Parallel Applications and Computing Technologies (IMPACT). His research interests include high-performance computing, parallel computer architectures, high-performance I/O, reconfigurable computing, experimental performance evaluations, computer vision, and remote sensing. He has authored or coauthored more than 200 refereed research papers and book chapters in these areas and his research has been supported by DoD/DARPA, NASA, NSF, and also industry, including IBM and SGI. He is the first author of the book *UPC: Distributed Shared Memory Programming*, which has the first formal specification of the UPC language used in high-performance computing. He is a member of the ACM and the Phi Kappa Phi National Honor Society, he was also a U.S. Fulbright Scholar, the recipient of the Alexander Schwarzkopf Prize for Technological Innovations and the recipient of the Alexander von Humboldt research award from the Humboldt Foundation in Germany.