Towards Area-Efficient Optical Neural Networks: An FFT-based Architecture

Jiaqi Gu[1*], Zheng Zhao[1], Chenghao Feng[1], Mingjie Liu[1], Ray T. Chen[1], and David Z. Pan[1†]

[1]ECE Department, The University of Texas at Austin
[*]jqgu@utexas.edu; [†]dpan@ece.utexas.edu

***Abstract*—As a promising neuromorphic framework, the optical neural network (ONN) demonstrates ultra-high inference speed with low energy consumption. However, the previous ONN architectures have high area overhead which limits their practicality. In this paper, we propose an area-efficient ONN architecture based on structured neural networks, leveraging optical fast Fourier transform for efficient computation. A two-phase software training flow with structured pruning is proposed to further reduce the optical component utilization. Experimental results demonstrate that the proposed architecture can achieve 2.2∼3.7× area cost improvement compared with the previous singular value decomposition-based architecture with comparable inference accuracy.**

I. Introduction

Deep neural networks (DNNs) have demonstrated superior performance in a variety of intelligent tasks, for example convolutional neural networks on image classification [1] and recurrent neural networks on language translation [2]. Multi-layer perceptrons (MLPs) are among the most fundamental components in modern DNNs, which are typically used as regression layers, classifiers, embedding layers, and attention layers, etc. However, MLPs require computationally-expensive matrix-vector multiplication, which becomes challenging for traditional von Neumann computing schemes owing to speed and energy inefficiency. To resolve this issue, significant efforts have been made on hardware design of neuromorphic computing frameworks to improve the computational speed of neural networks, such as electronic architectures [3], [4], [5] and photonic architectures [6], [7]. Among extensive neuromorphic computing systems, optical neural networks (ONNs) distinguish themselves by ultra-high bandwidth, ultra-low latency, and near-zero energy consumption. Even though ONNs are currently not competitive in terms of area cost, they still offer a promising alternative approach to microelectronic implementations given the above advantages.

Recently, several works demonstrated that MLP inference can be efficiently performed at the speed of light with optical components, e.g., spike processing [6] and reservoir computing [8]. They claimed a photodetection rate over 100 GHz in photonic networks, with near-zero energy consumption if passive photonic components are used [9]. Based on matrix singular value decomposition (SVD) and unitary matrix parametrization [10], [11], Shen *et al.* [12] designed and fabricated a fully optical neural network that achieves an MLP with Mach-zehnder interferometer (MZI) arrays. Once the weight matrices in the MLP are trained and decomposed, thermo-optic phase shifters on the arms of MZIs can be set up accordingly. Since the weight matrices are fixed after training, this fully optical neural network can be completely passive, thus minimizes the total energy consumption. However, this SVD-based architecture is limited by high photonic component utilization and area cost. Considering a single fully-connected layer with an $m \times n$ weight matrix, the SVD-based ONN architecture requires $\mathcal{O}(m^2+n^2)$ MZIs for implementation. Another work [13] proposed a slimmed ONN architecture (TΣU) based on the previous one [12], which substitutes one of the unitary blocks with a sparse tree network. However, its area cost improvement is limited. Therefore, this high hardware complexity of the SVD-based ONN architecture has become the bottleneck of its hardware implementation.

In addition to hardware implementation, recent advances in neural architecture design and network compression techniques have shown significant reduction in computational cost. For example, structured neural networks (SNNs) [14] were proposed to significantly reduce computational complexity and thus, become amenable to hardware. Besides, network pruning offers another powerful approach to slimming down neural networks by cutting off insignificant neuron connections. While non-structured pruning [15] produces random neuron sparsity, group sparsity regularization [16] and structured pruning [4] can lead to better network regularity and hardware efficiency. However, readily-available pruning techniques are rather challenging to be applied to the SVD-based architecture due to some issues, such as accuracy degradation and hardware irregularity. The gap between hardware-aware pruning and the SVD-based architecture gives another motivation for a pruning-friendly ONN architecture.

In this paper, we propose a new ONN architecture that improves area efficiency over previous ONN architectures. It leverages optical fast Fourier transform (OFFT) and its inverse (OIFFT) to implement structured neural networks, achieving lower optical component utilization. It also enables the application of structured pruning given its architectural regularity. The proposed architecture partitions the weight matrices into block-circulant matrices [17] and efficiently performs circulant matrix multiplication through OFFT/OIFFT. We also adopt a two-phase software training flow with structured pruning to further reduce photonic component utilization while maintaining comparable inference accuracy to previous ONN architectures. The main contributions of this work are as follows:

- We propose a novel, area-efficient optical neural network architecture with OFFT/OIFFT.
- We exploit a two-phase software training flow with structured pruning to learn hardware-friendly sparse neural networks that directly eliminate part of OFFT/OIFFT modules for further area efficiency improvement.
- We experimentally show that pruning is challenging to be applied to previous ONN architectures due to accuracy loss and retrainability issues.
- We experimentally demonstrate that our proposed architecture can lead to an area saving of 2.2∼3.7× compared with the previous SVD-based ONN architecture, with negligible inference accuracy loss.

The remainder of this paper is organized as follows. Section II introduces the background knowledge for our proposed architecture. Section III demonstrates the challenges to apply pruning to SVD-based architectures. Section IV presents details about the proposed architecture and software pruning flow. Section V analytically compares our hardware utilization with the SVD-based architecture. Section VI reports the experimental results on MNIST dataset [18] for our proposed architecture, followed by the conclusion in Section VII.

978-1-7281-4123-7/20/$31.00

II. Preliminaries

In this section, we introduce the background knowledge for our proposed architecture. We discuss principles of cirulant matrix representation and its fast computation algorithms in Section II-A and illustrate structured pruning techniques with Group Lasso regularization in Section II-B.

A. FFT-based Circulant Matrix Computation

Unlike the SVD-based ONNs which focus on classical MLPs, our proposed architecture are based on structured neural networks (SNNs) with circulant matrix representation. SNNs are a class of neural networks that are specially designed for computational complexity reduction, whose weight matrices are regularized using the composition of structured sub-matrices [14]. Among all structured matrices, circulant matrices are often preferred in recent SNN designs. As an example, we show an $n \times n$ circulant matrix $\boldsymbol{W}$ as follows,

$$\begin{bmatrix} w_0 & w_{n-1} & \cdots & w_1 \\ w_1 & w_0 & \cdots & w_2 \\ \vdots & \vdots & \ddots & \vdots \\ w_{n-1} & w_{n-2} & \cdots & w_0 \end{bmatrix}.$$

The first column vector $\boldsymbol{w} = [w_0, w_1, \ldots, w_{n-1}]^T$ represents all independent parameters in $\boldsymbol{W}$, and other columns are just its circulation.

According to [17], circulant matrix-vector multiplication can be efficiently calculated through fast Fourier transform. Specifically, given an $n \times n$ circulant matrix $\boldsymbol{W}$ and a length-n vector $\boldsymbol{x}$, $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ can be efficiently performed with $\mathcal{O}(n \log n)$ complexity as,

$$\boldsymbol{y} = \mathcal{F}^{-1}\big(\mathcal{F}(\boldsymbol{w}) \odot \mathcal{F}(\boldsymbol{x})\big), \tag{1}$$

where $\mathcal{F}(\cdot)$ represents n-point real-to-complex fast Fourier transform (FFT), $\mathcal{F}^{-1}(\cdot)$ represents its inverse (IFFT), and $\odot$ represents complex vector element-wise multiplication.

SNNs benefit from high computational efficiency while maintaining comparable model expressivity to classical NNs. Theoretical analysis [19] shows that SNNs can approximate arbitrary continuous functions with arbitrary accuracy given enough parameters, and are also capable of achieving the identical error bound to that of classical NNs. Therefore, based on SNNs with circulant matrix representation, the proposed architecture features low computational complexity and comparable model expressivity.

B. Structured Pruning with Group Lasso Penalty

The proposed ONN architecture enables the application of structured pruning to further save optical components, while maintaining accuracy and structural regularity. Structured pruning trims the neuron connections in NNs to mitigate computational complexity. Unlike ℓ_1 or ℓ_2 norm regularization, which produces arbitrarily-appearing zero elements, structured pruning with Group Lasso regularization [4], [20] leads to zero entries in groups. This coarse-grained sparsity is more friendly to hardware implementation than non-structured sparsity. The formulation of Group Lasso regularization term is given as follows,

$$\boldsymbol{L}_{GL} = \sum_{g=0}^{G} \sqrt{1/p_g}\, \|\beta_g\|_2\,, \tag{2}$$

where G is the total number of parameter groups, β_g is the parameter vector in the g-th group, $\|\cdot\|_2$ represents ℓ_2 norm, p_g represents the vector length of β_g, which accounts for the varying group sizes. Intuitively, the ℓ_2 norm penalty $\|\beta_g\|_2$ encourages all elements in the g-th group to converge to 0, and the group-wise summation operation is equivalent to group-level ℓ_1 norm regularization, which contributes to the coarse-grained sparsity. Leveraging the structured pruning together with Group Lasso regularization, our proposed architecture can save even more photonic components.

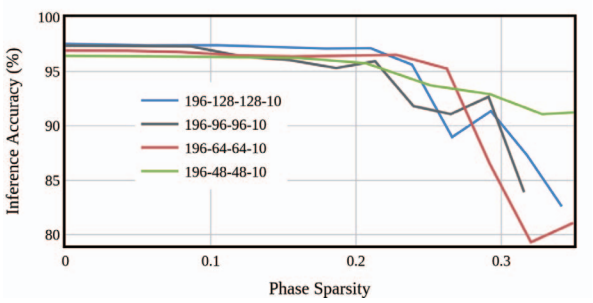

Fig. 1: Correlation between phase sparsity and inference accuracy with different network configurations based on MNIST [18] dataset.

III. Challenges in Pruning SVD-based Architecture

In this section, we demonstrate that the network pruning is experimentally challenging in the SVD-based architecture. As far as we know, it is hard to find any pruning method that can be directly applied to sparsifying MZI arrays.

In the SVD-based architecture, an $m \times n$ weight matrix $\boldsymbol{W}$ can be decomposed into $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$ using singular value decomposition. Unitary matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ can be further parametrized [10] into the product of planar rotation matrices $\boldsymbol{U} = \boldsymbol{D} \cdot \Pi_{i=m}^{2} \Pi_{j=1}^{i-1} \boldsymbol{R}_{ij}$, where $\boldsymbol{D}$ is a diagonal matrix and each unitary rotation $\boldsymbol{R}_{ij}$ can be represented by an angle or phase ϕ. Each unitary rotation matrix with phase ϕ can be implemented with an MZI. We denote all phases after parametrization as $\boldsymbol{\Phi}$. Phases with particular values, i.e., 0, $\pi/2$, π, and $-\pi/2$, can physically eliminate the use of the corresponding MZIs. We refer to these particular phases as sparse phases. One of the methods to perform pruning is to train a sparse weight matrix, but the sparsity can barely maintain after decomposition and parametrization. Another straight-forward method is post-training phase pruning. It directly clamps sparse phases but could cause significant accuracy degradation due to its unretrainability.

We experimentally illustrate the correlation between inference accuracy and phase sparsity. Concretely, we clamp $\boldsymbol{\Phi}$ with a threshold ϵ to get $\widehat{\boldsymbol{\Phi}}$. Then we evaluate the inference accuracy with reconstructed weight matrix $\widehat{\boldsymbol{W}}$. Figure 1 shows that, in different network configurations, on average no more than 15% phases can be pruned to achieve negligible ($\sim 0.5\%$) absolute accuracy degradation. With over 20% phases being pruned, its model expressivity will be severely harmed with significant accuracy loss ($> 1\%$). This accuracy loss partially attributes to the difficulty of retraining the weight matrices while maintaining phase sparsity. Another challenge derives from the hardware irregularity caused by non-structured phase pruning, which further limits the area improvement it can achieve. The above limitations also apply to the TΣU-based architecture [13] as it has a similar architectural design to the SVD-based one. This unsatisfying incompatibility between previous ONN architectures and pruning techniques offers a strong motivation for us to propose a new architecture to better leverage pruning techniques.

IV. Architecture and Structured Pruning

In this section, we will discuss details about the proposed architecture and pruning method. In the first part, we illustrate five stages of our proposed architecture. In the second part, we focus on the two-phase software training flow with structured pruning.

A. Proposed Architecture

Based on structured neural networks, our proposed architecture implements a structured version of MLPs with circulant matrix

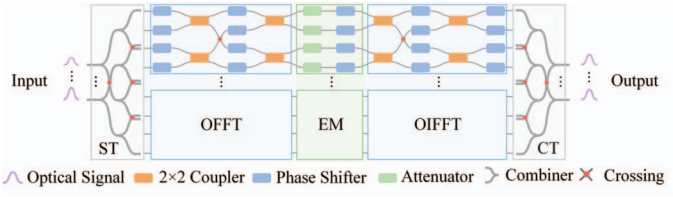

Fig. 2: Schematic diagram of a single layer of the proposed architecture. All adjacent phase shifters on the same waveguide are already merged into one phase shifter.

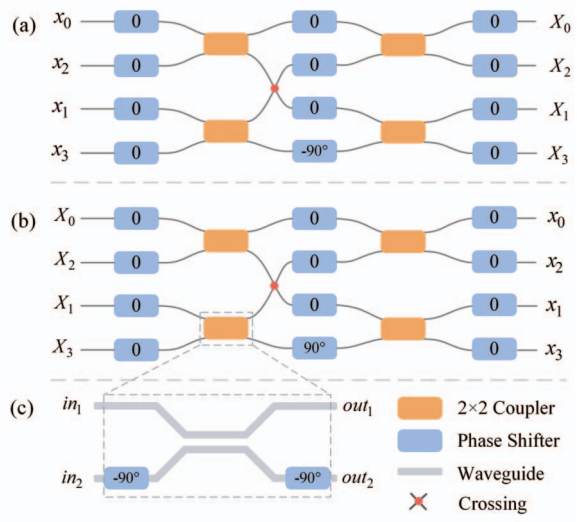

Fig. 3: Schematics of (a) 4-point OFFT, (b) 4-point OIFFT, and (c) 2×2 coupler. Note that phase shifters shown above are not merged for structural completeness consideration.

representation. A single layer in the proposed architecture performs linear transformation via block-circulant matrix multiplication $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$. Consider an n-input, m-output layer, the weight matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ is partitioned into $p \times q$ sub-matrices, each being a $k \times k$ circulant matrix. To perform tiled matrix multiplication, the input $\boldsymbol{x}$ is also partitioned into q segments $\boldsymbol{x} = (\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{q-1})$. Thus $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ can be performed in a tiled way,

$$\boldsymbol{y} = \begin{pmatrix} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_{p-1} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{q-1} \boldsymbol{W}_{0j}\boldsymbol{x}_j \\ \sum_{j=0}^{q-1} \boldsymbol{W}_{1j}\boldsymbol{x}_j \\ \vdots \\ \sum_{j=0}^{q-1} \boldsymbol{W}_{p-1j}\boldsymbol{x}_j \end{pmatrix}. \tag{3}$$

The i_{th} segment $y_i = \sum_{j=0}^{q-1} \boldsymbol{W}_{ij}\boldsymbol{x}_j$ is the accumulation of q independent circulant matrix multiplications. Each $\boldsymbol{W}_{ij}\boldsymbol{x}_j$ can be efficiently calculated using the fast computation algorithm mentioned in Eq. (1). Based on the aforementioned equations, we realize block-circulant matrix multiplication $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ in five stages: 1) Splitter tree (ST) stage to split input optical signals for reuse; 2) OFFT stage to calculate $\mathcal{F}(\boldsymbol{x})$; 3) element-wise multiplication (EM) stage to calculate $\mathcal{F}(\boldsymbol{w}_{ij}) \odot \mathcal{F}(\boldsymbol{x}_j)$ as described in Eq. (1); 4) OIFFT stage to calculate $\mathcal{F}^{-1}(\cdot)$; 5) combiner tree (CT) stage to accumulate partial multiplications to form the final results. $\mathcal{F}(\boldsymbol{w}_{ij})$ can be precomputed and encoded into optical components, thus there is no extra stage to physically perform it. The schematic diagram of our proposed architecture is shown in Fig. 2. Details of the above five stages will be discussed in the rest of this section.

1) OFFT/OIFFT Stages: To better model the optical components used to implement the OFFT/OIFFT stages, we introduce a unitary

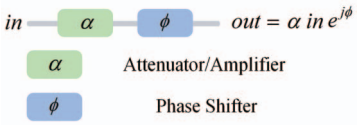

Fig. 4: Complex number multiplication realized by cascaded attenuator/amplifier and phase shifter.

FFT as,

$$\boldsymbol{X}_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \boldsymbol{x}_n e^{-i\frac{2\pi kn}{N}} \quad k = 0, 1, \cdots, N-1. \tag{4}$$

We denote this special operation as $\widehat{\mathcal{F}}(\cdot)$ and its inverse as $\widehat{\mathcal{F}}^{-1}(\cdot)$, to distinguish from the original FFT/IFFT operations. Equivalently, we re-write the circulant matrix multiplication with the above new operations,

$$\boldsymbol{y} = \widehat{\mathcal{F}}^{-1}\big(\mathcal{F}(\boldsymbol{w}) \odot \widehat{\mathcal{F}}(\boldsymbol{x})\big). \tag{5}$$

This unitary FFT operation can be realized with optical components. We first give a simple example for the optical implementation of a 2-point unitary FFT. As shown in Eq. (6), the transformation matrix of a 2-point unitary FFT can be decomposed into three transform matrices. They can be directly mapped to a 3-dB directional coupler with two $-\pi/2$ phase shifters on its lower input/output ports. For brevity, we refer to this cascaded structure as a 2×2 coupler, which is shown in Fig. 3(c).

$$\begin{aligned} \begin{pmatrix} out_1 \\ out_2 \end{pmatrix} &= \frac{1}{\sqrt{2}} \begin{pmatrix} in_1 + in_2 \\ in_1 - in_2 \end{pmatrix} \\ &= \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & -j \end{pmatrix}}_{\text{output phase shifter}} \underbrace{\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & j \\ j & 1 \end{pmatrix}}_{\text{directional coupler}} \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & -j \end{pmatrix}}_{\text{input phase shifter}} \begin{pmatrix} in_1 \\ in_2 \end{pmatrix} \end{aligned} \tag{6}$$

Based on 2×2 couplers and phase shifters, larger-sized OFFT/OIFFT can be constructed with a butterfly structure. The schematics of a simple 4-point OFFT and OIFFT are shown in Fig. 3(a) and Fig. 3(b). Extra 0-degree phase shifters are inserted for phase tuning purpose. This butterfly-structured OFFT may have scalability issues because the number of waveguide crossings (CR) will increase rapidly when the number of point gets larger. However, this unsatisfying scalability will not limit our proposed architecture for two reasons. First, only small values of k, e.g., 2, 4, 8, will be adopted to balance hardware efficiency and model expressivity. Second, input and output sequences can be reordered to avoid unnecessary waveguide crossings, as shown in Fig. 3.

2) EM Stage: In the EM stage, complex vector element-wise multiplications will be performed in the Fourier domain as $I_1 e^{j\phi_1} \cdot I_2 e^{j\phi_2} = I_1 \cdot I_2 e^{j(\phi_1+\phi_2)}$, where I and ϕ are intensity and phase of Fourier light signals respectively. Leveraging the polarization of light, we use optical attenuators (AT) or amplification materials/optical on-chip amplifiers to realize modulus multiplication $I_1 \cdot I_2$ and phase shifters for argument addition $e^{j(\phi_1+\phi_2)}$, which is shown in Fig. 4.

3) ST/CT Stage: We introduce tree-structured splitter/combiner networks to realize input signal splitting and output signal accumulation, respectively. To reuse input segments $\boldsymbol{x}_j$ in multiple blocks, optical splitters (SP) are used to split optical signals. Similarly, to accumulate partial multiplication results, i.e., $\boldsymbol{y}_i = \sum_{j=0}^{q-1} \boldsymbol{W}_{ij}\boldsymbol{x}_j$, we adopt optical combiners (CB) for signal addition. Given that optical splitters can be realized by using combiners in an inversed direction, we will focus on the combiner tree structure for brevity. The transfer function of an N-to-1 optical combiner is,

$$out = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} in_l. \tag{7}$$

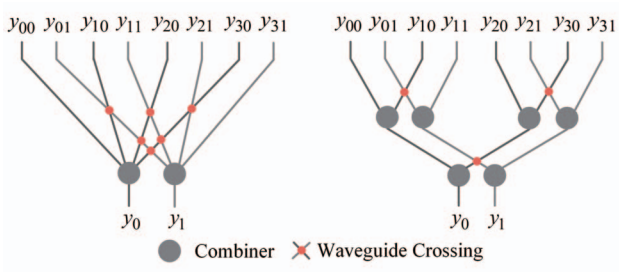

Fig. 5: Comparison between direct combining (left) and combiner tree (right) with 4 length-2 vectors accumulated.

Accumulating q length-k vectors by simply using k q-to-1 combiners introduces a huge number of waveguide crossings which may cause intractable implementation difficulty. Also, combiners with more than two ports are still challenging for manufacturing. In order to alleviate this problem, we adopt a tree-structured combiner network, shown in Fig. 5. This combiner tree consists of $k(q-1)$ combiners and reduces the number of waveguide crossings to $k(k-1)(q-1)/2$. Given that combiners will cause optical intensity loss by a factor of $1/\sqrt{N}$ as shown in Eq. (7), we assume there will be optical amplifiers added to the end to compensate this loss.

B. Two-phase Training Flow with Structured Pruning

Structured pruning can be applied to our proposed architecture during training given its architectural regularity. As described in Alg. 1, we exploit a two-phase software training flow with structured pruning to train a more compact NN model with fewer redundancies and negligible accuracy loss. Lines 2-4 perform the first initial training phase with Group Lasso regularization term added to our loss function.

$$\boldsymbol{L} = \boldsymbol{L}_{base} + \lambda\, \boldsymbol{L}_{GL}, \tag{8}$$

where $\boldsymbol{L}_{base}$ is the basic loss function, e.g., cross-entropy loss if targeted at classification tasks, and λ is a hyper-parameter used to weigh the regularization term $\boldsymbol{L}_{GL}$ given in Eq. 2. The initial training phase explores a good local minimum in the full parameter space to get rough convergence. This is designed to provide a good initial model for the subsequent pruning. Line 5 enters the structured pruning phase. The pruning mask $\boldsymbol{M}$ is generated to mark $\boldsymbol{w}_{ij}$ whose ℓ_2 norm falls below a threshold T. Those marked weight groups will be forced to zero. Hence, the corresponding hardware modules can be completely eliminated. As training and pruning are alternately performed, the network sparsity will incrementally improve. Line 12 applies a smooth function, e.g., polynomial or Tanh, to gradually increase pruning threshold to avoid accuracy degradation caused by aggressive pruning.

Algorithm 1 Two-Phase Training Flow with Structured Pruning

Require: Initial parameter $\boldsymbol{w}^0 \in \mathbb{R}^{p\times q\times k}$, pruning threshold T, initial training timestep t_{init}, and learning rate α;
Ensure: Converged parameter $\boldsymbol{w}^t$ and a pruning mask $\boldsymbol{M} \in \mathbb{Z}^{p\times q}$;
1: $\boldsymbol{M} \leftarrow 1$ ▷ Initialize pruning mask to all 1
2: **for** $t \leftarrow 1, ..., t_{init}$ **do** ▷ Phase 1: Initial training
3: $L^t(\boldsymbol{w}^{t-1}) \leftarrow L^t_{base}(\boldsymbol{w}^{t-1}) + \lambda \cdot L^t_{GL}(\boldsymbol{w}^{t-1})$
4: $\boldsymbol{w}^t \leftarrow \boldsymbol{w}^{t-1} - \alpha \cdot \nabla_{\boldsymbol{w}} L^t(\boldsymbol{w}^{t-1})$
5: **while** $\boldsymbol{w}^t$ not converged **do** ▷ Phase 2: Structured pruning
6: **for** all $\boldsymbol{w}^{t-1}_{i,j} \in \boldsymbol{w}^{t-1}$ **do**
7: **if** $||\boldsymbol{w}^{t-1}_{ij}||_2 < T$ **then**
8: $\boldsymbol{M}[i,j] \leftarrow 0$ ▷ Update pruning mask
9: `ApplyDropMask`$(\boldsymbol{M}, \boldsymbol{w}^{t-1})$
10: $L^t(\boldsymbol{w}^{t-1}) \leftarrow L^t_{base}(\boldsymbol{w}^{t-1}) + \lambda \cdot L^t_{GL}(\boldsymbol{w}^{t-1})$
11: $\boldsymbol{w}^t \leftarrow \boldsymbol{w}^{t-1} - \alpha \cdot \nabla_{\boldsymbol{w}} L^t(\boldsymbol{w}^{t-1})$
12: `UpdateThreshold`(T) ▷ Smoothly increase threshold

V. Theoretical Analysis on Proposed Architecture

In this section, we derive a theoretical estimation of hardware utilization of the proposed architecture and the SVD-based architecture. By comparing the hardware component utilization, we show that theoretically our proposed architecture costs fewer optical components than the SVD-based architecture. For simplicity, we convert all area-costly components, i.e., 2×2 couplers, MZIs, and attenuators, to 3-dB directional couplers (DCs) and phase shifters (PSs). Specifically, one 2×2 coupler can be taken as one DC and two PSs, and one MZI can be taken as two DCs and one PS. Since an attenuator can be achieved by a single-input directional coupler with appropriate transfer factor, we count one attenuator as one DC.

Given an n-input, m-output layer, the SVD-based implementation requires $m(m-1)/2+n(n-1)/2$ MZIs and $\max(m,n)$ attenuators to realize the weight matrix. Therefore, with the aforementioned assumption, the total number of components it costs is given by,

$$\begin{aligned} \#DC_{\text{SVD}} &= m(m-1) + n(n-1) + \max(m,n) \\ \#PS_{\text{SVD}} &= m(m-1)/2 + n(n-1)/2. \end{aligned} \tag{9}$$

For our architecture, each $k\times k$ circulant matrix costs k attenuators and corresponding components required by k-point OFFT/OIFFT. The following formulation gives the number of components for a k-point OFFT/OIFFT.

$$\begin{aligned} \#DC_{\text{OFFT}}(k) &= 2\times \#DC_{\text{OFFT}}(k/2) + k/2 = \frac{k}{2}\log_2 k \\ \#PS_{\text{OFFT}}(k) &= k(\log_2 k + 1) \end{aligned} \tag{10}$$

A phase shift is physically meaningful only when it is within $(-2\pi, 0]$ as phases can wrap around. Hence, multiple successive phase shifters on the same segment of a waveguide can be merged as one phase shifter, which can be seen when comparing Fig. 2 and Fig. 3. Then the total number of components used in our architecture to implement an $m \times n$ weight matrix with size-k circulant sub-matrices is given by,

$$\begin{aligned} \#DC_{\text{Ours}}(k) &= \frac{m}{k}\times\frac{n}{k}\times(2\times \#DC_{\text{OFFT}}(k) + k) \\ &= \frac{mn}{k}(\log_2 k + 1) \\ \#PS_{\text{Ours}}(k) &= \frac{m}{k}\times\frac{n}{k}\times(2\times \#PS_{\text{OFFT}}(k) - k) \\ &= \frac{mn}{k}(2\log_2 k + 1). \end{aligned} \tag{11}$$

In practical cases, k will be set to small values, such as 2, 4, and 8. Given arbitrary values of m and n, the proposed architecture costs theoretically fewer optical components than the SVD-based architecture.

VI. Experimental Results

We conduct numerical simulations for functionality validation and evaluate our proposed architecture on the hand-written digit recognition dataset (MNIST) [18] with various network configurations. Quantitative evaluation shows that our proposed architecture outperforms the SVD-based and TΣU-based ONN architectures in terms of area cost without any accuracy degradation.

A. Simulation Validation

To validate the functionality of our proposed architecture, we conduct optical simulations on a 4×4 circulant matrix-vector multiplication module using Lumerical INTERCONNECT tools. First we encode a 4×4 identity weight matrix into our architecture, and input 4 parallel optical signals to validate its functionality. For brevity, we plot several different representative cases in Fig. 6a. It shows that our designed architecture can correctly realize identity projection. Further, we randomly generate a length-4 real-valued weight vector

TABLE I: Comparison of inference accuracy and hardware utilization on MNIST dataset with different configurations.

	Network Configurations	Sparsity	#Parameter	Accuracy	#DC	#PS	Area (cm^2)
Model 1	SVD [12]: (28×28)-400-10	0.00	318 K	98.49%	934 K	467 K	20.62
	TΣU [13]: (28×28)-400-10	0.00	318 K	98.49%	777 K	388 K	17.15
	Ours w/o Prune: (28×28)-1024(8)-10(2)	0.00	105 K	98.32%	412 K	718 K	9.33
	Ours w/ Prune: (28×28)-1024(8)-10(2)	0.40	63 K	98.26%	244 K	425 K	5.53
Model 2	SVD [12]: (14×14)-70-10	0.00	14 K	96.93%	48 K	24 K	1.07
	TΣU [13]: (14×14)-70-10	0.00	14 K	96.93%	44 K	22 K	0.97
	Ours w/o Prune: (14×14)-256(4)-10(2)	0.00	14 K	96.93%	40 K	67 K	0.90
	Ours w/ Prune: (14×14)-256(4)-10(2)	0.45	8 K	96.91%	22 K	36 K	0.49
Model 3	SVD [12]: (28×28)-400-128-10	0.00	366 K	98.58%	967 K	483 K	21.35
	TΣU [13]: (28×28)-400-128-10	0.00	366 K	98.58%	794 K	396 K	17.52
	Ours w/o Prune: (28×28)-1024(8)-128(4)-10(2)	0.00	134 K	98.53%	501 K	868 K	11.34
	Ours w/ Prune: (28×28)-1024(8)-128(4)-10(2)	0.39	81 K	98.43%	289 K	517 K	6.77
Model 4	SVD [12]: (14×14)-160-160-10	0.00	59 K	97.67%	141 K	70 K	3.10
	TΣU [13]: (14×14)-160-160-10	0.00	59 K	97.67%	91 K	45 K	2.00
	Ours w/o Prune: (14×14)-256(4)-256(8)-10(2)	0.00	22 K	97.67%	73 K	123 K	1.64
	Ours w/ Prune: (14×14)-256(4)-256(8)-10(2)	0.37	14 K	97.52%	47 K	79 K	1.05

For example, configuration (28×28)-1024(8)-10(2) indicates a 2-layer neural network, where the first layer has 784 input channels, 1024 output channels with size-8 circulant matrices, and so on.

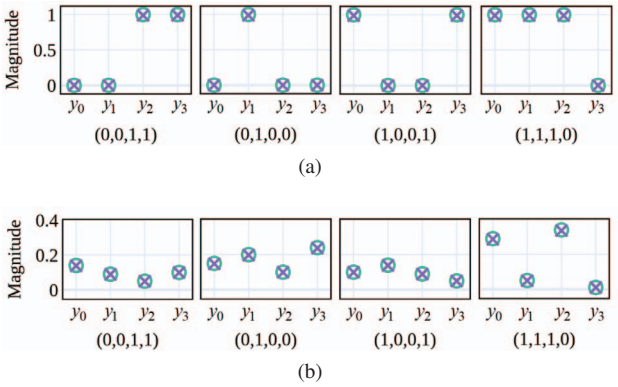

Fig. 6: (a) Simulated output intensities (crosses) and ground truth (circles) of a 4×4 identity circulant matrix-vector multiplication. (b) Simulated output intensities (crosses) and ground truth (circles) of a 4×4 circulant matrix-vector multiplication, with $\boldsymbol{w}$=(0.2,-0.1,0.24,-0.15). E.g., (0,0,1,1) is the input signal.

TABLE II: Optical component sizes used in the area estimation.

Optical Component	Length (μm)	Width (μm)
3-dB Directional Coupler [12]	54.4	40.3
Thermo-optic Phase Shifter [21]	60.16	0.50
2-to-1 Optical Combiner [22]	20.00	3.65
Waveguide Crossing [23]	5.9	5.9

$\boldsymbol{w} = (0.2, -0.1, 0.24, -0.15)$ to represent a circulant matrix, and encode $\mathcal{F}(\boldsymbol{w}) = (0.19e^{0j}, 0.064e^{-2.246j}, 0.69e^{0j}, 0.064e^{2.246j})$ into attenuators and phase shifters in the EM stage. The simulation results in Fig. 6b show good fidelity ($< 1.2\%$ maximum relative error) to the ground truth results.

B. Comparison Experiments

To evaluate the effectiveness and efficiency of our proposed architecture, we conduct a comparison experiment on a machine learning dataset MNIST [18], and compare the hardware utilization, model expressivity among four architectures: 1) SVD-based architecture [12]; 2) TΣU-based architecture [13]; 3) Ours without pruning; 4) Ours with pruning. For the SVD-based ONN, we simply train an original MLP since this architecture directly implements matrix multiplication in the fully-connected layer. In the TΣU-based architecture, training is performed on a sparse matrix T designed for dimensionality matching, a diagonal matrix Σ, and a pseudo-unitary matrix U with unitary regularization. This architecture eliminates one of the area-cost unitary matrix and adopts a sparse-tree T to match

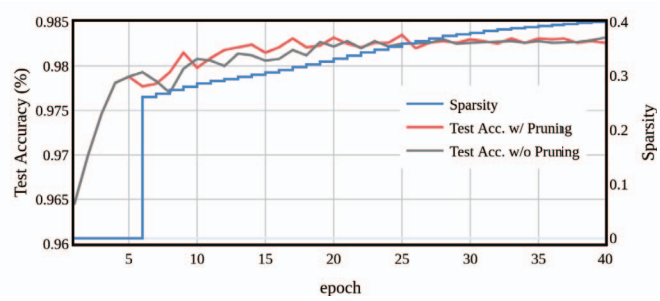

Fig. 7: Training curve of the proposed architecture with setup of (28×28)-1024(8)-10(2).

dimensionality. The orthogonality constraint of U is first relaxed with unitary regularization in the training, and satisfied by post-training unitary projection [13].

We implement the proposed architecture with different configurations in PyTorch and test the inference accuracy on a machine with an Intel Core i9-7900X CPU and an NVIDIA TitanXp GPU. We set λ to 0.3 for the Group Lasso regularization term, initialize all trainable weights with a Kaiming-Normal initializer [24], adopt the Adam optimizer [25] with initial learning rate=1×10^{-3} and a step-wise exponential-decay learning rate schedule with decay rate=0.9. We use the ideal rectified linear units (ReLU) activation function as nonlinearity. All NN models are trained for 40 epochs with mini-batch size of 32 till fully converged. Figure 7 plots the test accuracy and sparsity curves as training proceeds. The first 5 epochs are the initial training phase. After that, the model has roughly converged. In the subsequent structured pruning phase, we apply a step-wise function to smoothly increase the threshold T. We can see that every time sparsity increases, the test accuracy decreases accordingly and then fully recovers during the next training epoch. This alternate pruning and re-training mechanism incrementally improves sparsity while minimizing accuracy loss.

For fair comparison, all architectures are trained with the same hyper-parameters and have similar test accuracy in each experiment configuration. To estimate the component utilization and area cost, we adopt exactly the same type of photonic devices in all architectures, as listed in Table II, and accumulate the area of each optical component for approximation. Placement or routing information is not considered in our estimation. In Table I, the first column indicates different neural network configurations. For example, (14×14)-256(4)-10(2) describes a 2-layer network, with 196 input channels, 256 output channels in the first layer (k=4), and 10 output channels in the second layer (k=2). The TΣU-based architecture adopts a unique

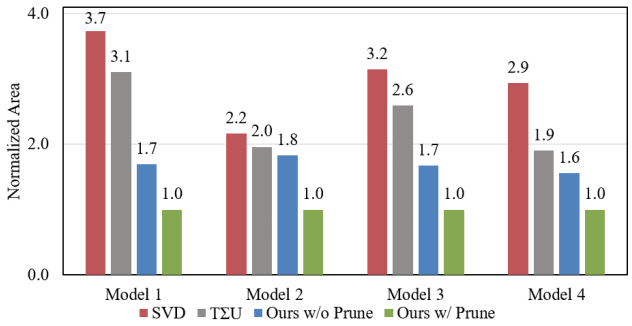

Fig. 8: Normalized area comparison with different model configurations. *Model 1-4* refer to Table I. *SVD* refers to [12] and $T\Sigma U$ refers to [13].

training methodology and claims to have small accuracy degradation ($< 1\%$) [13], thus we assume it has approximately the same accuracy as the SVD-based architecture. In the TΣU-based architecture, the total number of MZIs used to implement an $m \times n$ weight matrix is bounded by $n(n+1)/2$.

Among various network configurations, our proposed architecture outperforms the SVD-based architecture and the TΣU-based architecture with lower optical component utilization and better area cost. We normalize all areas to our architecture with pruning applied and show the normalized area comparison in Fig. 8. Consistent with analytical formulations in Section V, the experimental results show that, as the difference between input and output channels for each layer in the original MLPs gets larger, our proposed architecture can save a larger proportion of optical components. Specifically, fully-connected layers with highly-imbalanced input and output channels, e.g., ((28×28)-400-10), could benefit the most by using our proposed architecture. For MLPs with nearly-square weight matrices, e.g., ((14×14)-160-160-10), although the area gap is decreasing, our proposed architecture still shows superior area efficiency. This is because FFT-based structured matrix multiplications can reduce much parameter redundancies and save components while still maintaining model expressivity. Furthermore, ablation experiments on our structured pruning method validate the effectiveness of the proposed two-phase training flow. It can save an extra 30-50% optical components with negligible model expressivity loss. Even though the area cost is generally not where optical neuromorphic systems excel, their ultra-low latency and low power consumption make them very promising NN inference accelerators, e.g., in data centers. Therefore, by introducing an area-efficient and pruning-compatible ONN architecture, our work enables more compact ONN implementations without accuracy degradation.

VII. Conclusion

In this work, we propose an area-efficient optical neural network architecture. Our proposed ONN architecture leverages block-circulant matrix representation and efficiently realizes matrix-vector multiplication via optical fast Fourier transform. This architecture consists of five stages, including splitter tree, OFFT, element-wise multiplication, OIFFT, and combiner tree. Compared with the previous SVD-based and TΣU-based ONN architectures, the new design cuts down the optical component utilization and improves area cost by 2.2∼3.7× among various network configurations. We also propose a two-phase training flow to perform structured pruning to our architecture and further improve hardware efficiency with negligible accuracy degradation.

Acknowledgment

The authors acknowledge the Multidisciplinary University Research Initiative (MURI) program through the Air Force Office of Scientific Research (AFOSR), contract No. FA 9550-17-1-0071, monitored by Dr. Gernot S. Pomrenke.

References

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012.
[2] T. Mikolov, M. Karafiát, L. Burget *et al.*, "Recurrent neural network based language model," in *INTERSPEECH*, 2010.
[3] S. K. Esser, P. A. Merolla, J. V. Arthur *et al.*, "Convolutional networks for fast, energy-efficient neuromorphic computing," *PNAS*, 2016.
[4] Y. Wang, W. Wen, B. Liu *et al.*, "Group scissor: Scaling neuromorphic computing design to large neural networks," in *Proc. DAC*, 2017.
[5] Y. Zhang, X. Wang, and E. G. Friedman, "Memristor-based circuit design for multilayer neural networks," *IEEE TCAS I*, 2018.
[6] A. N. Tait, M. A. Nahmias, B. J. Shastri *et al.*, "Broadcast and weight: An integrated network for scalable photonic spike processing," *J. Light. Technol.*, 2014.
[7] J. Bueno, S. Maktoobi, L. Froehly *et al.*, "Reinforcement learning in a large-scale photonic recurrent neural network," *Optica*, 2018.
[8] D. Brunner, M. C. Soriano, C. R. Mirasso *et al.*, "Parallel photonic information processing at gigabyte per second data rates using transient states," *Nature Communications*, 2013.
[9] L. Vivien, A. Polzer, D. Marris-Morini *et al.*, "Zero-bias 40gbit/s germanium waveguide photodetector on silicon," *Opt. Express*, 2012.
[10] M. Reck, A. Zeilinger, H. Bernstein *et al.*, "Experimental realization of any discrete unitary operator," *Physical review letters*, 1994.
[11] A. Ribeiro, A. Ruocco, L. Vanacker *et al.*, "Demonstration of a 4×4-port universal linear circuit," *Optica*, 2016.
[12] Y. Shen, N. C. Harris, S. Skirlo *et al.*, "Deep learning with coherent nanophotonic circuits," *Nature Photonics*, 2017.
[13] Z. Zhao, D. Liu, M. Li *et al.*, "Hardware-software co-design of slimmed optical neural networks," in *Proc. ASPDAC*, 2019.
[14] Z. Li, S. Wang, C. Ding *et al.*, "Efficient recurrent neural networks using structured matrices in fpgas," in *ICLR Workshop*, 2018.
[15] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proc. NIPS*, 2015.
[16] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *Journal of Computational and Graphical Statistics*, 2013.
[17] O. Grandstrand, *Innovation and Intellectual Property Rights*. Oxford University Press, 2004.
[18] Y. LeCun, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/, 1998.
[19] L. Zhao, S. Liao, Y. Wang, Z. Li, J. Tang, and B. Yuan, "Theoretical properties for neural networks with weight matrices of low displacement rank," in *Proc. ICML*, 2017.
[20] J. Friedman, T. Hastie, and R. Tibshirani, "A note on the group lasso and a sparse group lasso," *arXiv preprint arXiv:1001.0736*, 2010.
[21] N. C. Harris, Y. Ma, J. Mower, T. Baehr-Jones, D. Englund, M. Hochberg, and C. Galland, "Efficient, compact and low loss thermo-optic phase shifter in silicon," *Opt. Express*, 2014.
[22] Z. Sheng, Z. Wang, C. Qiu, L. Li, A. Pang, A. Wu, X. Wang, S. Zou, and F. Gan, "A compact and low-loss mmi coupler fabricated with cmos technology," *IEEE Photonics Journal*, 2012.
[23] Y. Zhang, A. Hosseini, X. Xu, D. Kwong, and R. T. Chen, "Ultralow-loss silicon waveguide crossing using bloch modes in index-engineered cascaded multimode-interference couplers," *Opt. Lett.*, 2013.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. ICCV*, 2015.
[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.