L^{2}ight: Enabling On-Chip Learning for Optical Neural Networks via Efficient *in-situ* Subspace Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Silicon-photonics-based optical neural network (ONN) is a promising hardware platform that could represent a paradigm shift in efficient AI with its CMOS-compatibility, flexibility, ultra-low execution latency, and high energy efficiency. *In-situ* training on the online programmable photonic chips is appealing but still encounters challenging issues in on-chip implementability, scalability, and efficiency. In this work, we propose a closed-loop ONN on-chip learning framework L^{2}ight to enable scalable ONN mapping and efficient *in-situ* learning. L^{2}ight adopts a three-stage learning flow that first calibrates the complicated photonic circuit states under challenging physical constraints, then performs photonic core mapping via combined analytical solving and zeroth-order optimization. A subspace learning procedure with multi-level sparsity is integrated into L^{2}ight to enable *in-situ* gradient evaluation and fast adaptation, unleashing the power of optics for real on-chip intelligence. Extensive experiments demonstrate our proposed L^{2}ight outperforms prior ONN training protocols with **3-order-of-magnitude** higher scalability and over **30**$\times$ better efficiency, when benchmarked on various models and learning tasks. This synergistic framework is the *first* scalable on-chip learning solution that pushes this emerging field from *intractable* to *scalable* and further to *efficient* for next-generation self-learnable photonic neural chips. From a co-design perspective, L^{2}ight also provides essential insights for hardware-restricted unitary subspace optimization and efficient sparse training. We open-source our framework at link.

1 Introduction

The escalating scales of deep learning models and datasets have brought increased demand for computing capacities in electronic processors. Stringent performance and efficiency constraints in practical applications raise a surging need to develop more efficient computing solutions. As a promising substitute for conventional electronics, the optical neural networks (ONNs) have attracted extensive research interests owing to their sub-nanosecond latency and attojoule/multiply-accumulate operation (MAC) energy efficiency [39, 6, 48, 38], shown in Figure 1(a).

However, robustness and trainability are still critical issues for photonic AI engines [54, 19, 56]. Due to the analog computing nature of ONNs, the photonic DNN model inevitably suffer from performance degradation or even complete malfunction [54, 56] with the existence of manufacturing errors, non-ideal device controls, and undesired circuit noises, shown in Figure 1(b). Though non-ideal effects can be simulated and considered during software training [54, 19] to improve noise tolerance, the variation simulation is physically inaccurate (especially with unknown process variations) and prohibitively expensive, shown in Figure 1(c).

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Recently, on-device training becomes an appealing trend towards adaptable and self-learning ONNs. However, training on photonic neural chips is non-trivial and much less explored than on conventional platforms. Prior work [53, 22, 18, 15] only demonstrated small prototypes, and their scalability and efficiency are rather limited. To push the limits of DNNs in optics, we propose an efficient three-stage learning framework L^{2}ight that consists of variation-agnostic identity calibration, alternate projection-based parallel mapping, and multi-level sparse subspace learning. The main contributions of this work are four-fold,

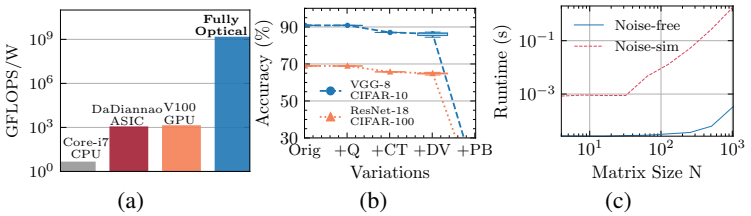


Figure 1: Comprehensive motivations. (a) Computational efficiency superiority of ONNs [39]. (b) Noise sensitivity of ONNs (Q: 8-bit quantization, CT: crosstalk, DV: device variation, PB: phase bias). (c) Runtime of noise-free matrix multiplication vs. w/ noise simulation (Q+CT+DV).

- **Scalability**. *For the first time*, an ONN learning protocol can scale up to million-level parameters under practical circuit non-ideality, over 3-order-of-magnitude more scalable than prior arts.
- **Efficiency**. We explore multi-level sparsity in *in-situ* gradient evaluation to trim down unnecessary on-chip training energy and runtime cost.
- **Learnability**. We show that by trading redundant representability, our restricted subspace optimization can provide ONNs with enough adaptability for on-device self-learning and task transfer.
- **Robustness**. Various practical device noises and process variations are considered *in situ* to facilitate noise-resilient photonic AI engines.
- To our best knowledge, this is the first framework that supports on-chip training on million-parameter ONNs, over **1000**× more scalable and **30**× more efficient than prior arts. Our PyTorch [35] library for ONN on-chip training is open-sourced at link.

2 Related Work

Optical Neural Network and Training Methods. One of recent ONN architectures adopts singular value decomposition (SVD) to implement matrix multiplication [39], i.e., $y = \boldsymbol{W}x = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*x$. Cascaded 2-by-2 optical devices, i.e., Mach-Zehnder interferometers (MZIs), are used to construct unitary matrices as the product of a series of 2-dimensional unitary rotators, $U(n) = D\prod_{i=n}^{2}\prod_{j=1}^{i-1} R_{ij}(\phi_{ij})$. A detailed introduction to ONNs can be found in Appendix A. Beyond offline training [19], ONN on-chip training methods are proposed to offload the process back onto photonics [22, 18, 15], shown in Table 1. Brute-force device tuning (BFT) [39, 55] and evolutionary algorithms [53] are applied to search MZI settings. An adjoint variable method (AVM) [22] is proposed to directly evaluate gradients using *in-situ* light field monitoring. Stochastic zeroth-order optimization (ZOO) [18, 15] is later applied to improve the training efficiency. However, prior methods are hard to scale to larger ONNs either due to algorithmic inefficiency or unrealistic hardware complexity.

Table 1: Scalability comparison with prior ONN on-chip training protocols in terms of #Params they can handle, used algorithm, resolution requirement (*Req.*), and circuit observability requirement. *Coh. I/O* is short for coherent input/output [30, 52]. ZO, FO mean zeroth- and first-order methods.

	BFT [39]	PSO [53]	AVM [22]	FLOPS [18]	MixedTrn [4]	L^{2}ight
#Params	∼100	∼100	∼100	∼1000	∼2500	**∼10 M**
Algorithm	ZO	ZO	FO	ZO	ZO	ZO+FO
Resolution Req.	Medium	High	Medium	High	Med	Medium
Observability Req.	Coh. I/O	Coh. I/O	Coh. I/O + Per device monitor	Coh. I/O	Coh. I/O	Coh. I/O

Efficient NN Training Methods. Extensive work has been devoted to accelerating DNN training, among which an important branch is sparse backpropagation. Previous methods mainly focus on approximating matrix multiplication by sparsifying the pre-activation gradients [41], forward and feedback matrices [1, 36], and input feature maps [34]. Quantization to the pre-activation gradients is

adopted in [49] to induce sparsity by trading off quantization steps and performance. Other methods also exist, e.g., distributed and low-precision training [2, 3, 23]. However, they are not readily applicable to analog photonic engines, thus not in the scope of our discussion.

Subspace Neural Networks. Subspace neural networks are special DNN models with restricted parameter space but demonstrate comparable representability to classical NNs. Sparse NNs [20, 47], low-rank NNs [9, 25, 42], structured NNs [11, 27, 45], Fourier-domain NNs [16, 32, 31], and general frequency-domain NNs [17] were introduced to trim down the redundancy in DNNs by restricting the NN structure, matrix rank, numerical resolution, etc. In this work, we deeply explore the trade-off between ONN learnability, trainability, and efficiency in the restricted unitary subspace.

Challenges of ONN On-Chip Training. As a unique hardware-restricted optimization problem, ONN *in-situ* learning encounters fundamental challenges causing scalability issues in prior methods:

- **Lack of full-observability for *in-situ* light field.** Tracking physical optical field on every waveguide in $\boldsymbol{U}$ and $\boldsymbol{V}^*$ is not scalable or practical when ONNs scale up. Per device light field monitoring and calibration [14, 22] involves intractable hardware complexity. In practice, only $\boldsymbol{\Sigma}$ can be precisely monitored and efficiently tuned.
- **Limited input/output observability**. In photonic tensor cores, for efficiency consideration, only the final output signals after $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$ can be coherently detected. Intermediate signals of a single unitary projection can not be easily read out without extra hardware support.
- **Inaccessible gradients for most control variables.** Due to the above two limitations, it is challenging to obtain true derivatives w.r.t. the MZI rotation phases in $\boldsymbol{U}$ and $\boldsymbol{V}^*$ [39, 18, 15], casting fundamental *in-situ* optimization difficulty as ONN scales up.

To enable *in-situ* self-learning for ONNs, the proposed synergistic framework `L²ight` provides a scalable, efficient, and on-chip-implementable solution that overcomes those hardware restrictions.

3 Synergistic ONN On-Chip Learning Framework `L²ight`

In this section, we give a formal description of the ONN on-chip training problem and detailed demonstration of our proposed three-stage learning flow `L²ight`, shown in Figure. 2.

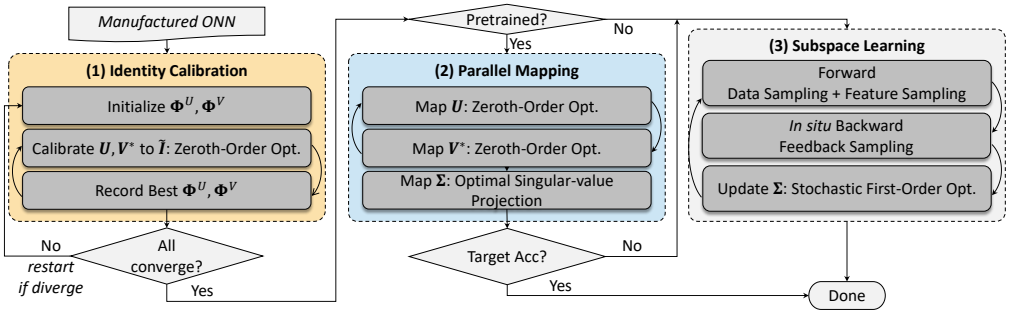


Figure 2: Proposed three-stage ONN on-chip learning flow `L²ight`.

3.1 Understanding the ONN On-Chip Learning Problem

The ONN that supports on-chip learning is shown in Figure 3, constructed by local storage, control units, interconnects, and photonic tensor cores with coherent I/O [30, 52] and wavelength-division multiplexing (WDM) [51, 43] for parallel processing. The target is to optimize MZI phases $\boldsymbol{\Phi}$ directly on chip under variations. Formally the *hardware-restricted* learning problem is,

$$
\begin{aligned}
&\boldsymbol{\Phi}^* = \underset{\boldsymbol{\Phi}}{\operatorname{argmin}}\ \mathcal{L}\big(\boldsymbol{W}(\boldsymbol{\Omega}\boldsymbol{\Gamma}\mathcal{Q}(\boldsymbol{\Phi}) + \boldsymbol{\Phi}_b); \mathcal{D}_{trn}\big),\\
\text{s.t. }\ &\boldsymbol{W}(\boldsymbol{\Phi}) = \big\{\boldsymbol{W}_{pq}(\boldsymbol{\Phi}_{pq})\big\}_{p=0,q=0}^{p=P-1,q=Q-1}, \quad \boldsymbol{W}_{pq}(\boldsymbol{\Phi}_{pq}) = \boldsymbol{U}_{pq}(\boldsymbol{\Phi}^U_{pq})\boldsymbol{\Sigma}_{pq}(\boldsymbol{\Phi}^S_{pq})\boldsymbol{V}^*_{pq}(\boldsymbol{\Phi}^V_{pq}),\\
&\boldsymbol{U}_{pq}(\boldsymbol{\Phi}^U_{pq}) = \boldsymbol{D}^U_{pq}\prod_{i=k}^{2}\prod_{j=1}^{i-1}\boldsymbol{R}_{pqij}(\phi^U_{pqij}), \quad \boldsymbol{V}^*_{pq}(\boldsymbol{\Phi}^V_{pq}) = \boldsymbol{D}^V_{pq}\prod_{i=k}^{2}\prod_{j=1}^{i-1}\boldsymbol{R}_{pqij}(\phi^V_{pqij}),\\
&\boldsymbol{\Sigma}_{pq}(\boldsymbol{\Phi}^S_{pq}) = \max(|\boldsymbol{\Sigma}_{pq}|)\texttt{diag}(\cdots, \cos\phi^S_{pq,i}, \cdots), \quad \boldsymbol{\Phi}_b \in \mathcal{U}(0, 2\pi),\ \boldsymbol{\Gamma} \in \mathcal{N}(\gamma, \sigma^2_\gamma).
\end{aligned}
\tag{1}
$$

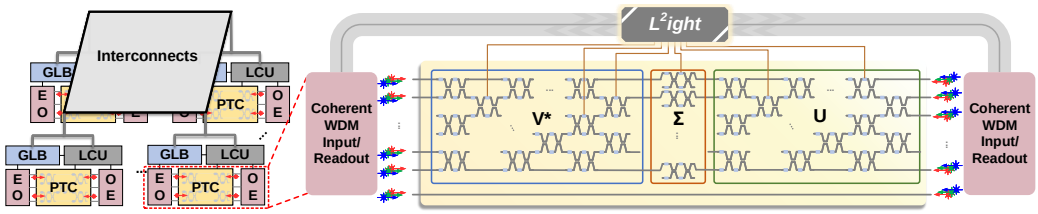


Figure 3: ONN architecture. PTC: photonic tensor core, GLB: global buffer, LCU: local control unit, EO: electrical-to-optical conversion.

The linear projection in an ONN adopts blocking matrix multiplication, where the $M \times N$ weight matrix is partitioned into $P \times Q$ blocks of size $k \times k$. During the optimization of $\mathbf{\Phi}$, we will jointly consider control resolution limits $\mathcal{Q}(\cdot)$ [19, 39], device process variations $\mathbf{\Gamma}$ [19, 18, 15], thermal crosstalk among adjacent devices $\mathbf{\Omega}$ [18, 56], and unknown phase bias due to manufacturing error $\mathbf{\Phi}_b$ for *in-situ* robustness-aware training. A detailed non-ideality analysis can be found in Appendix A.3.

3.2 Identity Calibration (IC): Variation-Agnostic Circuit State Preparation

After manufacturing, unknown process variations in waveguides make the initial state of PTCs unpredictable [44, 56]. A primary task is to prepare $\boldsymbol{U}$ and $\boldsymbol{V}^*$ to be identity matrices. However, the calibration problem, i.e., $\min_{\mathbf{\Phi}^U,\mathbf{\Phi}^V} \sum_{p,q} \left(\|\boldsymbol{U}_{pq}(\mathbf{\Phi}^U_{pq}) - \boldsymbol{I}\|_2^2 + \|\boldsymbol{V}^*_{pq}(\mathbf{\Phi}^V_{pq}) - \boldsymbol{I}\|_2^2\right)$, is not solvable given the observability and controllability constraints on $\boldsymbol{U}$ and $\boldsymbol{V}^*$. The closest auxiliary problem that we can solve is $\min_{\mathbf{\Phi}^U,\mathbf{\Phi}^V} \sum_{p,q} \left(\||\boldsymbol{U}_{pq}(\mathbf{\Phi}^U_{pq})| - \boldsymbol{I}\|_2^2 + \||\boldsymbol{V}^*_{pq}(\mathbf{\Phi}^V_{pq})| - \boldsymbol{I}\|_2^2\right)$. We denote those two mean square errors as MSE^U and MSE^V. We rewrite it as a surrogate minimization of $\mathcal{L}_{IC}$ that can lead to the same solution,

$$\min_{\mathbf{\Phi}} \sum_{p,q} \|\boldsymbol{U}_{pq}(\mathbf{\Phi}^U_{pq})\mathbf{\Sigma}_{pq}\boldsymbol{V}^*_{pq}(\mathbf{\Phi}^V_{pq})\mathbf{\Sigma}^{-1}_{pq} - I\|. \quad (2)$$

The optimal solution for this auxiliary problem is $\boldsymbol{U} = \boldsymbol{V}^* = \tilde{\boldsymbol{I}}$, where $\tilde{\boldsymbol{I}}$ is not guaranteed to be an identity matrix but a more general *sign-flipping matrix* with *arbitrary and unobservable sign flips* on the same columns in $\boldsymbol{U}$ and rows in $\boldsymbol{V}^*$, shown in Figure 4(a). We adopt zeroth-order optimization (ZOO) on $\mathbf{\Phi}^U$ and $\mathbf{\Phi}^V$ to calibrate $\boldsymbol{U}$ and $\boldsymbol{V}^*$ to approach $\tilde{\boldsymbol{I}}$, shown in Figure 4(b). We show the converged solution of Eq. (2) with unobservable sign flips and suboptimality only has marginal impacts on the following training procedure in later sections.

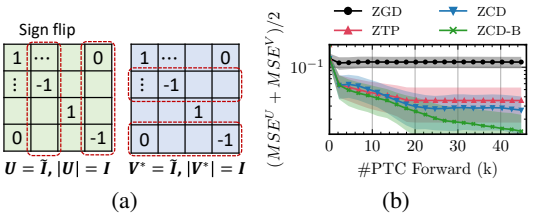


Figure 4: (a) Identity calibration with sign flip. (b) Different ZO optimizers on identity calibration. (ZGD: ZO gradient descent with momentum, ZCD: ZO coordinate descent, ZTP: ZO three-point. B is best solution recording.)

3.3 Parallel Mapping (PM): Alternate Projection-based Model Deployment

The target is to map the pre-trained weights $\boldsymbol{W}$ onto photonic MZI meshes $\widetilde{\boldsymbol{W}}(\mathbf{\Phi})$ with high fidelity. We formulate the parallel mapping as a batched $k \times k$-block-wise regression problem,

$$\min_{\mathbf{\Phi}} \sum_{p,q} \|\widetilde{\boldsymbol{W}}_{pq}(\mathbf{\Phi}_{pq}) - \boldsymbol{W}_{pq}\|_2^2. \quad (3)$$

As analyzed before, $\frac{\partial \boldsymbol{W}}{\partial \mathbf{\Phi}^U}$ and $\frac{\partial \boldsymbol{W}}{\partial \mathbf{\Phi}^V}$ are too expensive to compute *in situ*. We propose a parallel mapping flow with alternate zeroth-order optimization on $\mathbf{\Phi}^U$ and $\mathbf{\Phi}^V$. After convergence, we will perform analytical optimal singular-value projection (OSP) to minimize the regression error given fixed singular vectors.

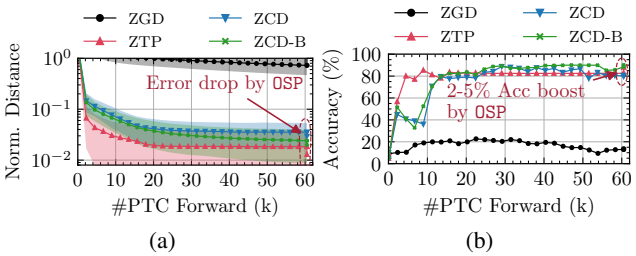


Figure 5: ZTP and ZCD-B perform the best in parallel mapping. The optimal singular-value projection leads to significant error drop and accuracy jump.)

We show why OSP gives the optimal solution under sign flips and how to perform it on the PTC.

Claim 1. *Optimal singular-value projection (OSP): the optimal singular value problem, i.e.,* $\boldsymbol{\Sigma}_{opt} = \text{argmin}_{\boldsymbol{\Sigma}} \|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^* - \boldsymbol{W}\|$*, can be analytically solved on-chip with arbitrary and unknown sign flip.*

Proof.

$$\boldsymbol{\Sigma}_{opt} = \texttt{diag}\big(\boldsymbol{U}^{-1}\boldsymbol{W}(\boldsymbol{V}^*)^{-1}\big) = \texttt{diag}\big(\boldsymbol{U}^*\boldsymbol{W}\boldsymbol{V}\big) = \texttt{diag}\big((\tilde{\boldsymbol{I}}^*\boldsymbol{V}^*\boldsymbol{W}^*\boldsymbol{U}\tilde{\boldsymbol{I}})^*\big). \tag{4}$$

OSP can be directly achieved using the limited operation set, i.e., $\{\boldsymbol{U}, \boldsymbol{U}^*, \boldsymbol{V}, \boldsymbol{V}^*\}$, supported by the reciprocal PTC itself. Specifically, we configure $\boldsymbol{V}^* = \tilde{\boldsymbol{I}}$ and $\boldsymbol{\Sigma} = \boldsymbol{I}$, and shine in a coherent WDM light beam that carries $\boldsymbol{W}$ from right ports. Since the coherent photonic circuit is reciprocal [30], we can read $\tilde{\boldsymbol{I}}\boldsymbol{U}^*\boldsymbol{W}$ on the left ports. Then we configure $\boldsymbol{U} = \tilde{\boldsymbol{I}}$ and $\boldsymbol{\Sigma} = \boldsymbol{I}$, and shine in its adjoint field from left, i.e., $\boldsymbol{W}^*\boldsymbol{U}\tilde{\boldsymbol{I}}^*$. We can directly read out the projected optimal diagonal on the right because the sign flips in the unitary matrices naturally cancel out on the diagonal. □

Figure 5 compares different ZO optimizers on this task. Coordinate-wise optimizers (ZCD [28] and ZTP [12]) outperform the gradient-based ZGD [13] with higher accuracy and convergence speed. This procedure is highly parallel and efficient since the mapping involves *no stochasticity* and only happens *locally* within each PTC. We can also observe that OSP effectively reduces the normalized matrix distance ($\|\boldsymbol{W} - \widetilde{\boldsymbol{W}}\|_2^2 / \|\boldsymbol{W}\|_2^2$) and boosts the accuracy by 2-5% almost for free.

3.4 Subspace Learning: Hardware-Aware Multi-Level Sparse Training

Besides mapping from an offline-trained model, L^{2}ight also supports *in-situ* self-learning fully on chip. We name this feature as *subspace learning*. To make L^{2}ight hardware-aware, we trade expensive full-space trainability for efficient subspace gradient evaluation, i.e., $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}}$ which coincides with the general frequency-domain ONNs [16, 17] and subspace NN design concept [40]. Since this learning stage involves stochasticity, it turns out to be the efficiency bottleneck, especially the backward pass. Hence, we explore multi-level sparsity for efficient *in-situ* gradient approximation.

3.4.1 *In-situ* Subspace Gradient Acquisition via Reciprocity in Optics

The conventional way to compute first-order gradients w.r.t. $\boldsymbol{\Sigma}$ is $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}} = \texttt{diag}\big(\boldsymbol{U}^* \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}} \boldsymbol{V}\big)$. However, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}} \boldsymbol{x}^T$ requires arbitrary matrix operations, which are not implementable by weight-stationary PTCs. Hence, we remap the gradient calculation as,

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}_\Sigma} \odot \boldsymbol{y}_V = \big(\tilde{\boldsymbol{I}}\boldsymbol{U}^* \frac{\partial \mathcal{L}}{\partial \boldsymbol{y}}\big) \odot (\tilde{\boldsymbol{I}}\boldsymbol{V}^*\boldsymbol{x}). \tag{5}$$

By shining in coherent WDM beams carrying the inputs and upstream gradients forward and backward through the reciprocal PTCs, respectively, as shown in Figure 6, the weight gradients can be efficiently obtained with lightweight element-wise multiplication $\odot$, which can be offloaded to electrical control units. Note that $\tilde{\boldsymbol{I}}$ naturally cancels out by the Hadamard product with no impacts on gradient fidelity.

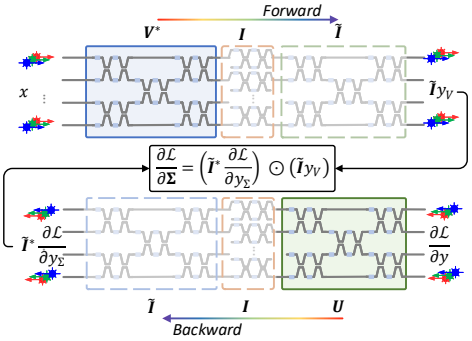
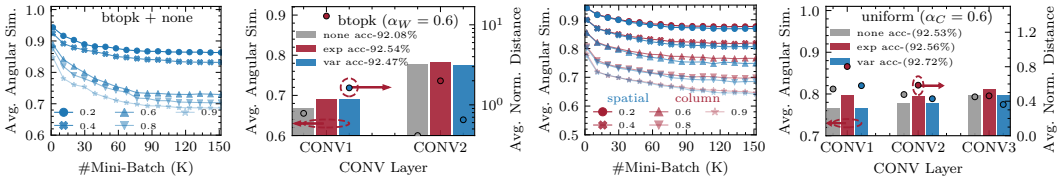


Figure 6: *In-situ* subspace gradient acquisition.

3.4.2 Multi-Level Sparse Subspace Learning

Inspired by sparse backpropagation methods [41, 34, 46, 33, 36], we propose multi-level sparse subspace learning to cut down both energy cost and total time steps in on-chip gradient evaluation.

Balanced Feedback Sampling. To improve the efficiency of the error feedback process, i.e., $\boldsymbol{W}^T \frac{\partial \mathcal{L}}{\partial y}$, as shown in Figure 7, we sample the feedback matrix $\boldsymbol{W}^T \in \mathbb{R}^{N \times M}$ with a structured sparse mask $\mathcal{P}_W = c_W(\mathcal{S}_W \otimes \mathbf{1})$ generated by the Kronecker product between a boolean mask $\mathcal{S}_W \in \{0,1\}^{Q \times P}$ with sparsity α_W and an all-ones matrix $\mathbf{1}$, where the scaling factor c_W is set to $\frac{1}{\alpha_W} = \frac{PQ}{\texttt{Tr}(\mathcal{S}_W^T \mathcal{S}_W)}$ for unbiased estimation, proven in Appendix D. The efficiency benefits come from two aspects: (1) the structurally masked PTCs are entirely idle, directly saving energy, and (2) the product accumulation depth/step is reduced by a factor of α_W, effectively trimming time steps.



(a) (b) (c) (d)

Figure 8: Average gradient angular similarity with different feedback sparsity (a) and three normalization methods (b). *none, exp,* and *var* represents no, expectation-maintained, and variance-maintained normalization. Average gradient angular similarity with spatial and column sampling (c) and three normalization methods (d).

However, two major downsides exist on traditional `uniform` and layer-wise `topk` sampling [36]. First, on a backward path, multiple feedback sampling operators will be cascaded, such that importance-unaware `uniform` sampling can lead to an exponentially large variance [34]. Second, `topk` sampling is overly greedy and tends to break the load balance as the feedback latency can be bottlenecked by the longest partial product accumulation path, shown in Figure 7. To tackle this, we propose a balanced top-K sampling (`btopk`) to draw $\mathcal{S}_W$ from a guided distribution that locally prefers blocks with large Frobenius norm, which can be efficiently evaluated by $\|\boldsymbol{W}_{pq}\|_{\mathcal{F}}^2 = \texttt{Tr}(|\boldsymbol{\Sigma}_{pq}|^2)$. It strikes a *balance between gradient variance and bias* by fine-grained row-wise top-K sampling and *eliminates load-imbalance* by guaranteeing the same sparsity for different rows of $\boldsymbol{W}^T$, i.e., $\sum_p \mathcal{S}_W(1,:) = \sum_p \mathcal{S}_W(2,:) = \cdots = \sum_p \mathcal{S}_W(Q,:)$. Figure 8(a), 8(b) shows the gradient approximation fidelity in terms of average angular similarity [5] and normalized matrix distance. Our `btopk`-sampled weight gradients align well with the true gradients. With the unbiased (`exp`) normalization factor α_W, `btopk` shows the best gradient angular similarity and inference accuracy compared with others.

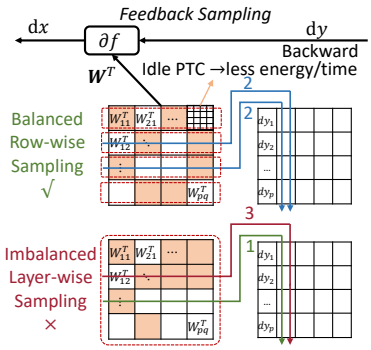


Figure 7: Balanced v.s. imbalanced feedback matrix sampling.

Information-Preserving Column Sampling. Input feature sparsification can also effectively cut down the gradient evaluation cost [36, 34], especially for costly CONV layers. However, with traditional *spatial sampling* (SS) [36, 34], the input feature map x barely maintains its sparsity regularity after being transformed to flattened patches X via *im2col* if the kernel size is larger than 1, shown in Figure 9. Hence, we propose a novel *column sampling* (CS) as a better solution. We sample X using a mask $\mathcal{S}_C\{0,1\}^{H'W'}$ with a uniform sparsity α_C, which is shared across batches with negligible overhead. This leads to both information preservation and efficiency improvement. First, in Figure 9, a pixel appears in multiple columns, such that partial information can be maintained after column sampling. Second, this highly-structured column dropping directly translates to less PTC forward energy and fewer partial gradient accumulation steps. In contrast, with a spatial mask $\mathcal{S}_S$ and spatial sparsity α_S, the masked pixel will be completely dropped with poor regularity after *im2col*, at the cost of large variance due to information loss and almost no runtime improvement on this dense linear projection engines. Note that for CONV1×1, CS turns out to be equivalent to SS, which can simultaneously save memory and runtime. Figures 8(c), 8(d) show that our proposed CS can obtain better gradient approximation fidelity than prior SS. Different normalization has small effects on model accuracy since feature sampling only happens locally within each layer, without any variance cascade effect. Note that simultaneous scaling by α_W and α_C tends to generate overly-confident gradient approximation, which empirically leads to harmful gradient variance. Hence, we will adopt α_C=1 in all experiments.

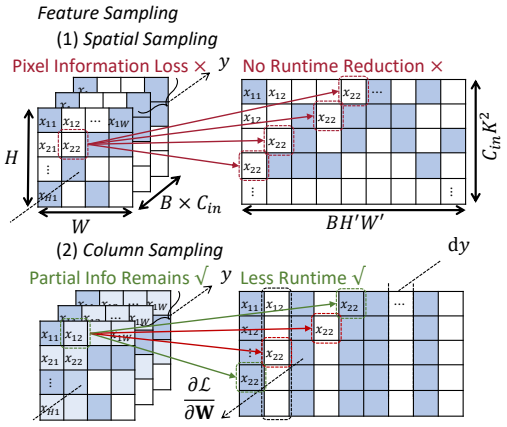


Figure 9: Spatial and column sampling for CONV.

Data Sampling. After parallel mapping, the ONN is initialized fairly close to the target pre-trained model. It is reasonable and intuitive to calibrate it with a representative calibration set instead of the entire training set. Inspired by the mini-batch dropping (SMD) technique [46], we integrate this SMD technique into our framework to further explore data-level sparsity. Within one training epoch, we randomly skip each iteration with probability α_D, directly translating to training time reduction.

4 Results

4.1 Experiment Setup

Datasets. We evaluate L^{2}ight on Vowel [10], MNIST [26], FashionMNIST [50], CIFAR-10, CIFAR-100 [24], and TinyImagenet [7]. On CIFAR-10/100 and TinyImagenet, we adopt random crop, flip, color jittering for augmentation.

Models. We evaluate on a customized MLP (8-16-16-4) [15] on Vowel, CNN-S (CONV8K3S2-CONV6K3S2-FC10) [15] on MNIST, a CNN-L ({CONV64K3}$\times$3-Pool5-FC10) on FashionMNIST, and VGG-8 [8]/ResNet-18 [1] [21] on CIFAR-10/100. CNN-L/FashionMNIST is used for ablation studies. VGG-8/ResNet-18 on CIFAR-10/100 are used for accuracy and efficiency comparison. Training details can be found in Appendix E.

Efficiency Evaluation. We assume fully parallel 9×9-blocking matrix multiplication in photonic tensor cores and sequential partial product accumulation in electronics. Our simulator counts the total number of PTC calls as the normalized energy indicator and the longest accumulation path as the normalized latency/runtime indicator. Details of profiling can be found in Appendix F.

4.2 Main Results

Scalability Comparison with Prior ONN Learning Protocols. Figure 10 compares L^{2}ight with two SOTA ONN on-chip training protocols, FLOPS [18] and MixedTrn [15]. Prior protocols can only handle toy models ($\sim$2,000 params) given their algorithmic inefficiency and instability, while our L^{2}ight shows $>$1,000$\times$ higher scalability to handle large ONNs ($\sim$10 M) on challenging tasks with comparable accuracy to full-space pre-trained models. Though MixedTrn achieves comparable accuracy to L^{2}ight on small benchmarks, we are still 1.7$\times$ faster and 6.9$\times$ more energy efficient.

The superiority of L^{2}ight provides three important insights: (1) *decoupling ZOO from stochasticity* and *partitioning a large-scale regression problem into a batch of sub-tasks* can greatly mitigate the curse of dimensionality both in convergence and efficiency. (2) *mapping before learning* can fully leverage the pre-trained model to reduce the learning cost. Prior methods have to learn from a severely corrupted solution under variations, while L^{2}ight recovers most accuracy via mapping, leaving a very light workload for subspace learning. (3) Restricted subspace learning provides *adequate degree of freedom* for training from scratch and task transfer. Also, its *compatibility with first-order* methods significantly boosts the trainability and breaks the scalability barrier for ONN training. We now validate the above insights by extensive experiments.

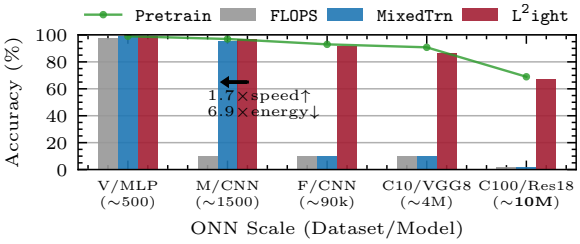


Figure 10: Compare scalability with prior protocols [18, 15].

Training Efficiency Comparison with Prior Sparse Training Methods. In Figure 11, we show accuracy and efficiency comparison of 1) baseline L^{2}ight-SL (BS), 2) L^{2}ight-SL with spatial sampling (RAD), 3) L^{2}ight-SL with weight and spatial sampling (SWAT-U), and 4) L^{2}ight-SL with all three introduced sampling methods (feedback, column, and data sampling), and 5) our proposed full flow with IC, PM, and sparse SL (L^{2}ight). When we perform subspace learning from scratch, our proposed sampling strategies outperform previous RAD and SWAT-U by $\sim$3$\times$ in hardware cost with comparable accuracy. Though RAD can save the forward peak memory, it leaves the most expensive backward pass unoptimized, which does not fully exploit the sparsity in ONN training. SWAT-U tries to save forward cost by simultaneously sparsifying the forward and feedback weights with shared

[1] https://github.com/kuangliu/pytorch-cifar

masks/pattern. However, in our experiment, the forward sparsification considerably degrades the model performance, which dilates the efficiency benefits from it. Our full three-stage flow L^{2}ight achieves the best accuracy assisted by parallel mapping, even higher than the non-sampled baseline, with over 30× higher energy efficiency and fewer time steps.

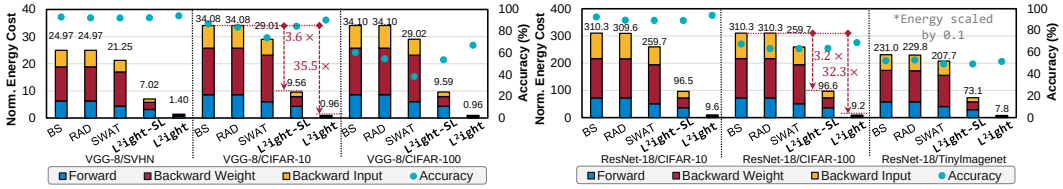
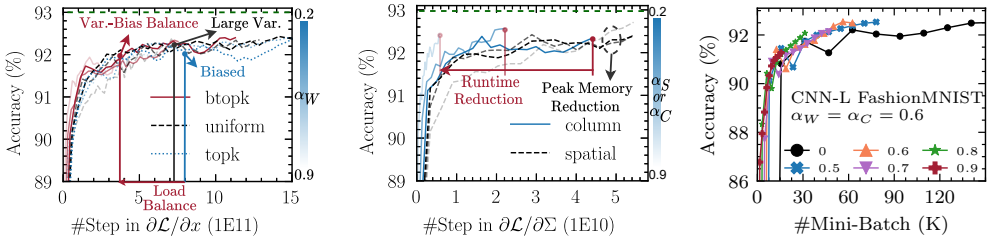


Figure 11: Accuracy and hardware efficiency comparison on VGG-8 (*left*) and ResNet-18 (*right*).

4.3 Ablation Studies and Discussion

4.3.1 Multi-Level Sparsity in Efficient Training

Feedback Sparsity. To investigate the impact of feedback sampling strategies, we visualize the gradient approximation fidelity and accuracy curves in Figure 12(a). `uniform` sampling shows varying performance under different sparsity values due to large gradient variances. `topk` shows worse performance after sufficient steps due to its biased gradient estimation from overly greedy block selection. In contrast, our proposed *load-balancing* `btopk` strikes a balance between variance and bias via block-wise sampling and also leads to less runtime as it forces load balance among massively parallel PTCs. In Table 2, feedback sampling saves 50-60% time steps on the most costly error feedback $\nabla_x \mathcal{L}$, leading to 1.5-1.8× overall time step reduction with minimum accuracy drop.



(a) (b) (c)

Figure 12: Accuracy v.s. weight gradient computation steps with three feedback sampling strategies (a) and different feature sampling techniques (b). Accuracy (93.02%) from a full-space trained model (green). CNN-L/FashionMNIST is used for (a) and (b). Compare different data sampling sparsity (c).

Feature Sparsity. Figure 12(b) compares the accuracy and weight gradient computation time steps on two feature sampling techniques. Though *spatial sampling* (`ss`) can save peak storage by dropping a subset of activations during the forward pass, it shows no gradient computation step reduction. Our hardware-friendly *column sampling* (`cs`) directly leads to energy and runtime reduction due to its

Table 2: Compare sampling strategies on CIFAR-10. Metrics are accuracy, activation size reduction, energy, and time step. Forward, weight gradient, and error feedback are denoted as $\mathcal{L}$, $\nabla_\Sigma \mathcal{L}$, and $\nabla_x \mathcal{L}$. L^{2}ight-SL is subspace learning from scratch, and L^{2}ight (IC→PM→SL) is the complete flow with non-ideal $\tilde{\boldsymbol{I}}$.

	Acc$_{\pm\sigma}$ (%)	Act↓(%)	Norm. PTC Energy				Norm. #Step			
			$\mathcal{L}$	$\nabla_\Sigma \mathcal{L}$	$\nabla_x \mathcal{L}$	Total (Ratio)	$\mathcal{L}$	$\nabla_\Sigma \mathcal{L}$	$\nabla_x \mathcal{L}$	Total (Ratio)
L^{2}ight-SL (Baseline) VGG-8	86.66$_{\pm 0.13}$	-	8.58	17.16	8.34	34.08 (1.00)	32.64	5.49	92.02	130.14 (1.00)
+ Feedback Sampling (α_W=0.6)	86.41$_{\pm 0.25}$	-	8.58	17.16	3.38	29.13 (1.17)	32.64	5.49	35.76	73.89 (1.76)
+ Column Sampling (α_C=0.6)	85.58$_{\pm 0.01}$	-	8.58	7.16	3.38	19.12 (1.78)	32.64	4.67	35.76	73.07 (1.78)
+ Data Sampling (α_D=0.5)	84.45$_{\pm 0.45}$	-	4.29	3.58	1.69	9.56 (3.56)	16.32	2.34	17.89	36.54 (3.56)
+ RAD [34] (α_S=0.85)	83.68$_{\pm 0.58}$	11.78	8.58	17.16	8.34	34.08 (1.00)	32.64	5.49	92.02	130.14 (1.00)
+ SWAT-U [36] (α_W=0.3, α_S=0.6)	73.91$_{\pm 0.27}$	8.31	6.01	17.16	5.84	29.01 (1.17)	25.98	5.49	82.19	113.66 (1.15)
L^{2}ight (IC→PM→SL)	**90.20**$_{\pm 0.05}$	-	**0.43**	**0.36**	**0.17**	**0.96 (35.64)**	**1.63**	**0.23**	**1.79**	**3.65 (35.64)**
L^{2}ight-SL (Baseline) ResNet-18	92.37$_{\pm 0.08}$	-	72.24	144.49	93.60	310.33 (1.00)	463.40	27.23	1,478.84	1,969.48 (1.00)
+ Feedback Sampling (α_W=0.5)	91.35$_{\pm 0.03}$	-	72.24	144.49	48.13	264.86 (1.17)	463.40	27.23	747.22	1,237.85 (1.59)
+ Column Sampling (α_C=0.5)	90.02$_{\pm 0.16}$	4.47	72.24	72.49	48.13	192.86 (1.61)	463.40	15.68	747.21	1,226.30 (1.61)
+ Data Sampling (α_D=0.5)	89.07$_{\pm 0.04}$	4.47	36.13	36.26	24.07	96.46 (3.22)	231.76	7.84	373.71	613.31 (3.21)
+ RAD [34] (α_S=0.9)	89.44$_{\pm 0.17}$	46.60	72.26	143.72	93.60	309.58 (1.00)	463.53	26.03	1,478.84	1,969.00 (1.00)
+ SWAT-U [36] (α_W=0.3, α_S=0.5)	89.21$_{\pm 0.16}$	25.89	50.57	143.64	65.52	259.73 (1.19)	358.40	26.56	1,417.96	1,802.00 (1.09)
L^{2}ight (IC→PM→SL)	**93.91**$_{\pm 0.02}$	4.47	**3.61**	**3.62**	**2.41**	**9.64 (32.20)**	**23.16**	**0.78**	**37.34**	**61.29 (32.13)**

structured sparsity. In Table 2, when column sampling is further added, we observe ∼50% PTC energy saving on weight gradient computation $\nabla_{\Sigma}\mathcal{L}$ at the cost of ∼1% accuracy drop.

Data Sparsity. In the data level, we also demonstrate how SMD with sparsity α_D impacts the training efficiency in Figure 12(c). With the best selected α_W and α_C, data sparsity directly reduces training time by skipping iterations [46]. The data sampling selects a uniform subset of the training set to represent the original data distribution, leading to less data processing with comparable generalization in the extracted features. Another explanation is that the variance increased by partial replacement serves as a regularization mechanism to improve the model performance [46]. For relatively easy tasks, aggressive sparsity (α_D=0.8) is a sweet point, while for larger datasets shown in Table 2, a medium sparsity (0.5) can be a good setting to balance both the training cost and accuracy. With all three sampling methods integrated, our L^{2}ight-SL shows competitive accuracy and ∼3× higher efficiency than RAD and SWAT-U. More advanced dataset sampling methods are left for future exploration.

4.3.2 Learnability of Restricted Subspace ONNs

Impacts of Calibration/Mapping Quality. Table 2 shows that with IC and PM, the full L^{2}ight flow achieves the highest accuracy with 32-35× efficiency boost over baselines. We further evaluate the impact of different mapping accuracy and the calibration quality on subspace learning in Figure 13. First, *the optimality on subspace bases matters* as it determines the upper bound of recovered accuracy via subspace learning. With roughly optimized space bases, i.e., $\boldsymbol{U}, \boldsymbol{V}^*$, subspace learning can efficiently train basis coefficients, i.e., $\boldsymbol{\Sigma}$, achieving 5-6% higher accuracy and 9.9× less energy and steps compared with random unitaries (train from scratch). Second, *subspace optimization shows low sensitivity on mapping quality* and is able to compensate for the suboptimality in singular vectors within a reasonable range. Even with 60% mapped accuracy, singular value optimization has enough capability to recover the accuracy to ∼90%. Third, our subspace learning is *robust to gradient noises* caused by non-ideal $\tilde{\boldsymbol{I}}$ ($MSE^U \approx MSE^V \approx 0.013$), which shows that L^{2}ight can tolerate reasonable suboptimality in the calibration and mapping stages.

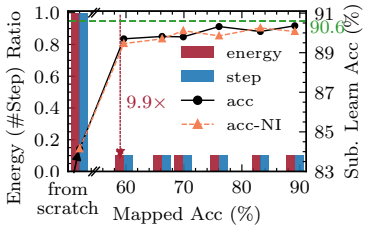


Figure 13: Impact of mapping accuracy (VGG-8 CIFAR-10 with α_W=α_C=0.6, α_D=0.5). *acc-NI* is the curve with non-ideal $\tilde{\boldsymbol{I}}$.

***In-situ* Transferability in the Restricted Subspace.** Another important question to answer is the transferability of subspace learning. After mapping, we fix the inherited unitaries and adapt to different tasks by only training the singular values. Figure 14 shows that the inherited bases span a good design space with enough transferability. The *in-situ* subspace transfer learning shows 1-2% higher final accuracy. Also, it uses 3∼5× fewer steps to obtain the same accuracy as training from scratch. Hence, our proposed L^{2}ight finds a highly trainable design point while the learnability is still mostly maintained.

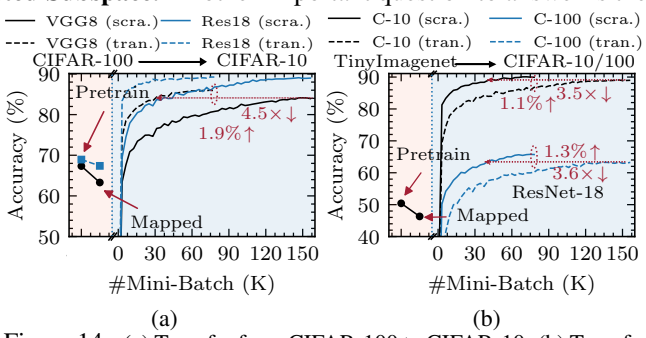


Figure 14: (a) Transfer from CIFAR-100 to CIFAR-10. (b) Transfer from TinyImagenet to CIFAR-10/100.

5 Conclusion

In this work, we propose the *first* scalable and efficient on-chip learning framework L^{2}ight for emerging optical neural networks. Our proposed three-stage flow synergistically enables on-chip self-learning via automatic circuit state calibration, parallel model mapping, and efficient subspace learning. To further improve the learning efficiency, we explore multi-level sparsity, including balanced feedback sampling, information-preserving column feature sampling, and runtime-reduced data sampling. Extensive ablation studies and comparison experiments show 3-order-of-magnitude scalability improvement over prior on-chip training protocols and 30× efficiency boost compared with previous sparse training methods.

References

[1] Menachem Adelman and Mark Silberstein. Faster neural network training with approximate tensor operations. *arXiv preprint arXiv:1805.08079*, 2018.

[2] Ron Banner, Itay Hubara, Elad Hoffer, and Daniel Soudry. Scalable methods for 8-bit training of neural networks. In *Proc. NeurIPS*, 2018.

[3] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. In *Proc. ICML*, volume 80, pages 560–569, 2018.

[4] Adel Bibi, El Houcine Bergou, Ozan Sener, Bernard Ghanem, and Peter Richtarik. A stochastic derivative-free optimization method with importance sampling: Theory and learning to control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[5] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[6] Q. Cheng, J. Kwon, M. Glick, M. Bahadori, L. P. Carloni, and K. Bergman. Silicon Photonics Codesign for Deep Learning. *Proceedings of the IEEE*, 2020.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, pages 248–255, 2009.

[8] Lei Deng, Peng Jiao, Jing Pei, Zhenzhi Wu, and Guoqi Li. GXNOR-Net: Training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework. *Neural Networks*, 2018.

[9] Emily Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation . In *Proc. NIPS*, 2014.

[10] D.H. Deterding. Speaker normalisation for automatic speech recognition. PhD thesis, University of Cambridge, 1989.

[11] Caiwen Ding, Siyu Liao, Yanzhi Wang, Zhe Li, Ning Liu, et al. CirCNN: Accelerating and Compressing Deep Neural Networks Using Block-Circulant Weight Matrices. In *Proc. MICRO*, pages 395–408, 2017.

[12] El Houcine Bergou and Eduard Gorbunov and Peter Richtárik. Stochastic Three Points Method for Unconstrained Smooth Minimization. *SIAM Journal on Optimization*, 2020.

[13] Saeed. Ghadimi and Guanghui. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 2013.

[14] Stefano Grillanda, Marco Carminati, Francesco Morichetti, et al. Non-invasive monitoring and control in silicon photonics using CMOS integrated electronics. *Optica*, 2014.

[15] Jiaqi Gu, Chenghao Feng, Zheng Zhao, Zhoufeng Ying, Ray T Chen, and David Z Pan. Efficient on-chip learning for optical neural networks through power-aware sparse zeroth-order optimization. In *Proc. AAAI*, 2021.

[16] Jiaqi Gu, Zheng Zhao, Chenghao Feng, et al. Towards area-efficient optical neural networks: an FFT-based architecture. In *Proc. ASPDAC*, 2020.

[17] Jiaqi Gu, Zheng Zhao, Chenghao Feng, et al. Towards Hardware-Efficient Optical Neural Networks: Beyond FFT Architecture via Joint Learnability. *IEEE TCAD*, 2020.

[18] Jiaqi Gu, Zheng Zhao, Chenghao Feng, Wuxi Li, Ray T. Chen, and David Z. Pan. FLOPS: Efficient On-Chip Learning for Optical Neural Networks Through Stochastic Zeroth-Order Optimization. In *Proc. DAC*, 2020.

[19] Jiaqi Gu, Zheng Zhao, Chenghao Feng, Hanqing Zhu, Ray T. Chen, and David Z. Pan. ROQ: A noise-aware quantization scheme towards robust optical neural networks with low-bit controls. In *Proc. DATE*, 2020.

[20] Song Han, Huizi Mao, and William Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *Proc. ICLR*, 2016.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, pages 770–778, 2016.

[22] Tyler W. Hughes, Momchil Minkov, Yu Shi, and Shanhui Fan. Training of photonic neural networks through in situ backpropagation and gradient measurement. *Optica*, 2018.

[23] Xianyan Jia, , Shutao Song, Wei He, Yangzihao Wang, Haidong Rong, Feihu Zhou, et al. Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes. *arXiv preprint arXiv:1807.11205*, 2018.

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[25] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lem. Speeding-up convolutional neural networks using fine-tuned cp-decomposition . In *Proc. ICLR*, 2015.

[26] Y. LeCun. The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/, 1998.

[27] Zhe Li, Shuo Wang, Caiwen Ding, et al. Efficient recurrent neural networks using structured matrices in fpgas. In *ICLR Workshop*, 2018.

[28] Xiangru Lian, Huan Zhang, Cho-Jui Hsieh, Yijun Huang, and Ji Liu. A Comprehensive Linear Speedup Analysis for Asynchronous Stochastic Parallel Optimization from Zeroth-Order to First-Order. In *Proc. NeurIPS*, 2016.

[29] Maziyar Milanizadeh, Douglas Aguiar, Andrea Melloni, and Francesco Morichetti. Canceling thermal cross-talk effects in photonic integrated circuits. *J. Light. Technol.*, 2019.

[30] David A.B. Miller. Analyzing and generating multimode optical fields using self-configuring networks. *Optica*, 2020.

[31] Mario Miscuglio, Zibo Hu, Shurui Li, Jonathan K. George, Roberto Capanna, Hamed Dalir, Philippe M. Bardet, Puneet Gupta, and Volker J. Sorger. Massively parallel amplitude-only fourier neural network. *Optica*, 7(12):1812–1819, Dec 2020.

[32] Mario Miscuglio, Zibo Hu, Shurui Li, Jiaqi Gu, et al. Million-channel parallelism Fourier-optic convolutional filter and neural network processor. In *Proc. CLEO*, 2020.

[33] Arild Nøkland. Direct Feedback Alignment Provides Learning in Deep Neural Networks. In *Proc. NIPS*, 2016.

[34] Deniz Oktay, Nick McGreivy, Joshua Aduol, Alex Beatson, and Ryan P Adams. Randomized automatic differentiation. In *Proc. ICLR*, 2021.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[36] Md Aamir Raihan and Tor M. Aamodt. Sparse weight activation training. In *Proc. NeurIPS*, 2020.

[37] M Reck, A Zeilinger, HJ Bernstein, et al. Experimental realization of any discrete unitary operator. *Physical review letters*, 1994.

[38] Bhavin J. Shastri, Alexander N. Tait, T. Ferreira de Lima, Wolfram H. P. Pernice, Harish Bhaskaran, C. D. Wright, and Paul R. Prucnal. Photonics for artificial intelligence and neuromorphic computing. *Nature Photonics*, 2021.

[39] Yichen Shen, Nicholas C. Harris, Scott Skirlo, et al. Deep learning with coherent nanophotonic circuits. *Nature Photonics*, 2017.

[40] Mengying Sun, Inci M. Baytas, Liang Zhan, Zhangyang Wang, and Jiayu Zhou. Subspace network: Deep multi-task censored regression for modeling neurodegenerative diseases. In *Proc. KDD*, page 2259–2268, 2018.

[41] Xu Sun, Xuancheng Ren, Shuming Ma, and Houfeng Wang. meProp: Sparsified Back Propagation for Accelerated Deep Learning with Reduced Overfitting. In *Proc. ICML*, 2017.

[42] Cheng Tai, Tong Xiao, Yi Zhang, Xiaogang Wang, and Weinan E. Convolutional neural networks with low-rank regularization . In *Proc. ICLR*, 2016.

[43] D. T. H. Tan, A. Grieco, and Y. Fainman. Towards 100 channel dense wavelength division multiplexing with 100ghz spacing on silicon. *Opt. Express*, 2014.

[44] Erman Timurdogan, Zhan Su, Christopher V. Poulton, et al. AIM Process Design Kit (AIMPDKv2.0): Silicon Photonics Passive and Active Component Libraries on a 300mm Wafer. In *Optical Fiber Communication Conference*, 2018.

[45] Yitu Wang, Fan Chen, Linghao Song, C.-J. Richard Shi, Hai Helen Li, and Yiran Chen. ReBoc: Accelerating Block-Circulant Neural Networks in ReRAM. In *Proc. DATE*, pages 1472–1477, 2020.

[46] Yue Wang, , Ziyu Jiang, Xiaohan Chen, Pengfei Xu, Yang Zhao, Yingyan Lin, and Zhangyang Wang. E2-Train: Training State-of-the-art CNNs with Over 80% Less Energy. In *Proc. NeurIPS*, 2019.

[47] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Proc. NIPS*, 2016.

[48] Gordon Wetzstein, Aydogan Ozcan, Sylvain Gigan, Shanhui Fan, Dirk Englund, Marin Soljačić, Cornelia Denz, , David A. B. Miller, and Demetri Psaltis. Inference in artificial intelligence with deep optics and photonics. *Nature*, 2020.

[49] Simon Wiedemann, Temesgen Mehari, Kevin Kepp, and Wojciech Samek. Dithered backprop: A sparse and quantized backpropagation algorithm for more efficient deep neural network training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

[50] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *Arxiv*, 2017.

[51] J. Yu and X. Zhou. Ultra-high-capacity dwdm transmission system for 100g and beyond. *IEEE Communications Magazine*, 2010.

[52] H. Zhang, M. Gu, X. D. Jiang, J. Thompson, H. Cai, S. Paesani, R. Santagati, A. Laing, Y. Zhang, M. H. Yung, Y. Z. Shi, F. K. Muhammad, G. Q. Lo, X. S. Luo, B. Dong, D. L. Kwong, L. C. Kwek, and A. Q. Liu. An optical neural chip for implementing complex- valued neural network. *Nature Communications*, 2021.

[53] Tian Zhang, Jia Wang, Yihang Dan, Yuxiang Lanqiu, Jian Dai, Xu Han, Xiaojuan Sun, and Kun Xu. Efficient training and design of photonic neural network through neuroevolution. *Optics Express*, 2019.

[54] Zheng Zhao, Jiaqi Gu, Zhoufeng Ying, et al. Design technology for scalable and robust photonic integrated circuits. In *Proc. ICCAD*, 2019.

[55] Hailong Zhou, Yuhe Zhao, Gaoxiang Xu, Xu Wang, Zhipeng Tan, Jianji Dong, and Xinliang Zhang. Chip-Scale Optical Matrix Computation for PageRank Algorithm. *JSTQE*, 2020.

[56] Ying Zhu, Grace Li Zhang, Bing Li, et al. Countering Variations and Thermal Effects for Accurate Optical Neural Networks. In *Proc. ICCAD*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Section 4.3.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] see Section 3.1
   (b) Did you include complete proofs of all theoretical results? [Yes] see Appendix D
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Please see Section 4.1, also add anonymous URL in paper Section 1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please see Section 4.1 and Appendix E.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Please see Table 2 with results from multiple runs.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see Appendix E.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes] We cited PyTorch.

   (b) Did you mention the license of the assets? [Yes] The license info is available online.

   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We include our ONN training library.

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]