ELight: Enabling Efficient Photonic In-Memory Neurocomputing with Life Enhancement

Hanqing Zhu*, Jiaqi Gu, Chenghao Feng, Mingjie Liu, Zixuan Jiang, Ray T. Chen, and David Z. Pan†
ECE Department, The University of Texas at Austin, Austin, TX, USA
*hqzhu@utexas.edu; †dpan@ece.utexas.edu

2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC) | 978-1-6654-2135-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ASP-DAC52403.2022.9712497

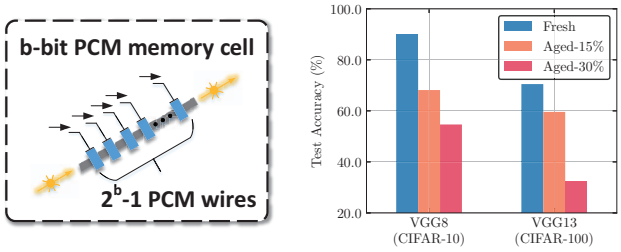
***Abstract*— With the recent advances in optical phase change material (PCM), photonic in-memory neurocomputing has demonstrated its superiority in optical neural network (ONN) designs with near-zero static power consumption, time-of-light latency, and compact footprint. However, photonic tensor cores require massive hardware reuse to implement large matrix multiplication due to the limited single-core scale. The resultant large number of PCM writes leads to serious dynamic power and overwhelms the fragile PCM with limited write endurance. In this work, we propose a synergistic optimization framework, `ELight`, to minimize the overall write efforts for efficient and reliable optical in-memory neurocomputing. We first propose write-aware training to encourage the similarity among weight blocks, and combine it with a post-training optimization method to reduce programming efforts by eliminating redundant writes. Experiments show that `ELight` can achieve over $20\times$ reduction in the total number of writes and dynamic power with comparable accuracy. With our `ELight`, photonic in-memory neurocomputing will step forward towards viable applications in machine learning with preserved accuracy, order-of-magnitude longer lifetime, and lower programming energy.**

I. Introduction

Optical neural networks (ONNs) [1]–[11] are widely studied as a promising neurocomputing paradigm with ultra-high speed, ultra-low energy cost, and high bandwidth to satisfy computation demands of machine learning applications. Recent work [12], [13] demonstrates that phase change material (PCM) can be used to build photonic tensor cores (PTCs) for optical in-memory matrix multiplication. PCM cells are programmable with non-volatile states as a weight encoding mechanism. By shining light through waveguides integrated with configured PCMs, light-matter interactions will change the amount of light transmission passively, thus achieving in-memory multiplication with near-zero static power. Moreover, the broadband transmission of PCMs enables massive parallelism with wavelength-division multiplexing (WDM). Holding above superiority, PCM-based PTCs open a new pathway towards efficient in-memory neurocomputing via photons.

However, PCM-based photonic in-memory neurocomputing still encounters practical barriers before truly viable for efficient inference acceleration. Firstly, current PCM cell designs support only limited bit-width data imprint. In [12], a reasonable implementation of b-bit PCM cell with low programming complexity is proposed for PTCs, with demonstrated $4\sim5$-bit storage. It is equivalently realized by patterning 2^b-1 identical PCM wires on the same waveguide, as shown in Fig. 1a. Each wire represents binary phase states by completely *crystallization* or *amorphization*, individually switched via *electrothermal heating*, thus allowing total 2^b nonlinear transmission levels. Hence, a specialized quantization strategy that suits the unique transmission level distribution is in high demand to preserve the ONN accuracy on the above low-precision PCM-based PTCs.

Besides limited bit-width, the potential frequent weight reprogramming during inference also raises critical issues in PCM-based photonic in-memory computing. Massive reuse of PTCs is required due to the limited scale of PTC compared to the weight matrix, e.g., a 64×64 PTC is already quite large. The resultant massive weight updates in PTCs potentially threaten PCM wires at a high risk of over-utilization, given limited write endurance of PCM, ranging from 10^6 to 10^8 [14]. Once PCM wires are aged and lose reprogrammability, the implementable transmission range will degrade, and the physical value will deviate from the desired value, leading to severe accuracy drops, shown in Fig. 1b. Moreover, the massive reprogramming has a non-trivial dynamic energy cost, which dilutes the energy efficiency benefits from PCM. The above issues are related to two key metrics: (1) The average utilization of PCM wires indicated by the total number of write operations (*# total writes*); (2) The maximum number (*# max writes*) of write operations over a single PCM cell in one PTC.



(a) (b)

Fig. 1: (a) Multi-level photonic memory cell based on PCM wires. (b) Accuracy comparison of models with/without a fixed ratio of aged cells.

In this work, we propose a synergistic aging-aware optimization framework `ELight` to tackle the issues. Based on an augmented redundant write elimination strategy, we devote to trimming down *redundant* writes on PCM wires in photonic memories during weight updates so as to reduce *# total writes* and *# max writes*. Aware of the block pattern of weight reloading, we first propose a write-aware training method to orchestrate the higher similarity among weight blocks to increase the eliminable redundant writes. Then post-training optimization is applied to reduce the number of writes further. The main contributions of this paper are listed as follows.

- **Distribution-Aware Quantization** scheme is introduced to reduce weight encoding errors on PCM cells with the awareness of modeled transmissivity distribution.
- **Write-Aware Training** is proposed to boost block-wise weight similarity and reduce redundant writes during weight updates with negligible effect on accuracy.
- **Post-Training Optimization** is employed to further cut down redundant writes via column-based reordering, without changing the model output.
- To the best of our knowledge, this is the *first work* that handles the aging and energy efficiency issue of PCM-based photonic neural engines. Our `ELight` achieves over $20\times$ reduction in the total number of re-programming operations and dynamic energy cost during inference, enabling long-life and efficient photonic in-memory neurocomputing.

978-1-6654-2135-5/22/$31.00 ©2022 IEEE

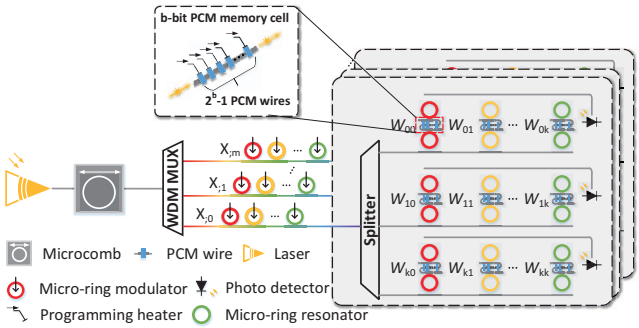


Fig. 2: The architecture of photonic tensor engines based on PCM.

II. Preliminaries

In this section, we introduce the basics of phase change material, the architecture of photonic tensor core, and current barriers in the practical deployment of PCM-based ONNs.

A. Basics of Phase Change Material (PCM)

As a promising memristive device, phase change material (PCM) draws particular attraction in photonic in-memory computing. PCM can cause a pronounced change in the optical property and switch between low-light-transmission crystalline (c) state, representing a logical '0', and high-light-transmission amorphous (a) state, representing a logical '1'. The programmable non-volatile states endow PCM with a potential to demonstrate ultra-fast in-memory multiplication [12], [13]. By shining light through the waveguide with PCM cells on top, the transmitted optical power can be modulated as $P_{out} = t \cdot P_{in}$ with t being the transmission factor inscribed in PCM states, where the scalar multiplication is implemented. Yet, the number of realized distinct transmission levels in current PCM cell designs is rather limited for high-precision data storage. In [12], an implementation of b-bit PCM cells with 2^b transmission levels is proposed for in-memory photonic analog computing by patterning 2^b-1 binary PCM wires on one waveguide. Considering the practicality of binary PCM devices and low programming complexity, we will focus on this PCM cell design [12] in the following discussion. However, PCM suffers from limited endurance with a maximum of $10^6 \sim 10^8$ total reprogramming times [14]. Frequent writing operations will over-utilize PCM, shorten its lifetime, and reduce reliability due to the loss of reprogrammability.

B. Architecture of Photonic Tensor Core

Recent work [12], [13] demonstrates the implementation of photonic tensor cores for optical in-memory matrix multiplication, i.e, $Y = WX + b$. Both utilize the photonic PCM arrays' storage and light interaction capability, with W being encoded in the PCM states.

Consider the weight block $W \in \mathbb{R}^{k \times k}$ and input matrix X with m columns. Fig. 2 illustrates one architecture of PCM-based PTC [12], where matrix-matrix multiplication (MM) is achieved by duplicating PCM array m times to carry out multiple matrix-vector multiplication (MVM). Starting with an input laser and an on-chip frequency comb to generate multiple wavelengths $(\lambda_0, \lambda_1, \cdots)$, a WDM multiplexer is used to distribute the light into m rows evenly. For each row, a series of narrowband micro-ring modulators are used to imprint one column of the input matrix X in the power of the optical signals. Then, a PCM array with k rows and k columns can be used to encode the weight block and achieve MVM between W and one column of X. Concretely, in the i-th rail, the inputs are filtered by the on-resonance micro-rings and weighted via light-matter interaction with the PCM. At the end of the drop port, photo-detectors are used to accumulate the intensity of the WDM optical signals, i.e., the weighted inputs, and output the desired result $Y_{im} = \sum_j^k W_{ij} X_{jm}$.

In Fig. 2, the b-bit PCM cell is equivalently realized by patterning 2^b-1 binary PCM wires on the same waveguide to enable 2^b transmission levels. To store the desired value, the photonic memory cell is programmed by selectively switching the phase of the wires between crystalline and amorphous states. For example, for a 4-bit photonic memory with fifteen PCM wires, '1100' is demonstrated by randomly programming twelve wires to the high-light-transmission amorphous state while setting others to the crystalline state. Note that the electrothermal a-c and c-a transition of PCM wires is achieved by sending electrical pulses to individual thermal heaters in parallel.

C. Barriers in Practical Deployment of PCM-based ONNs

As a promising platform for machine learning (ML), photonic in-memory neurocomputing still encounters practical challenges. First, the limited programming resolution of PCM-based PTCs calls for a specialized quantization scheme. Besides, the limited scale of PTC cannot promise the one-shot realization of large matrix multiplication. A 64×64 PTC is already quite large due to area cost and light loss [13]. Hence, massive reuse of PTCs is required during inference, leading to frequent weight updates on PCM arrays. Consequently, with limited endurance, PCM wires are at high risk of over-utilization, thus shortening the lifetime of photonic tensor cores. Moreover, a massive number of write operations requires significant dynamic power, which might raise concerns about the energy superiority claimed for PCM.

In this paper, to tackle the above issues, we trace and optimize two key metrics. The total number of write operations of PCM wires (*# total writes*) reflects the averaged degree of utilization of PCM wires and the dynamic energy cost. The maximum number of wire write operations of a single photonic cell (*# max writes*) can represent the status of the most over-utilized memory cell, determining the lifetime of PTC. Unlike previous works [15]–[18], which focus on optimizing frequent weight updates when the training is performed on emerging neuromorphic computing systems such as ReRAM, in this work, we focus on the potential frequent weight reprogramming during the inference process, which plagues photonic in-memory neurocomputing as a curse of the limited scale of PTCs.

III. Proposed Distribution-Aware Quantization Scheme

In this section, we give out a dedicated *distribution-aware quantization scheme* based on the analysis of transmission level distribution of b-bit PCM memory cell, to reduce weight encoding errors.

A. Transmission Model of Multi-Level PCM Memory Cell

The light transmissivity of a b-bit photonic memory cell is determined by the phase states of 2^b-1 PCM wires. Assuming that level i refers to the condition that i wires are set to c state while the others are set to a state, its extinction ratio (ER) is computed as the ratio of the transmitted optical power in level i and level 0, i.e., $10\log_{10}(\frac{P_i}{P_0})$. As demonstrated in [12], the ER uniformly increases as a function of i with a step Δe. Given that the transmitted optical power can be written as the product of the transmission factor t_i and input light power, ER can be further expressed as

$$10\log_{10}(\frac{t_i \cdot P_{in}}{t_0 \cdot P_{in}}) = 10\log_{10}(\frac{t_0 \cdot P_{in}}{t_0 \cdot P_{in}}) + i\Delta e. \quad (1)$$

Thus, the i-th transmission level can be derived as

$$t_i = t_0 \cdot 10^{\frac{1}{10}\Delta e \times i} = t_0 \times c^i,\ c = 10^{\frac{1}{10}\Delta e}. \quad (2)$$

Here, $c \in (0,1)$ indicates the percentage of light power transmitted through one PCM wire. The 0-th transmission level t_0 corresponds to all wires being in a state, which is approximately 1 [12].

Hence, we can finally formulate the distribution of 2^b transmission levels of b-bit PCM memory cell as an *exponential model*,

$$t_i = c^i, \ i = 0, 1, \ldots, 2^b - 1. \tag{3}$$

B. Augmented Base-c Quantization

An important question is how to effectively map full-precision weights w to the exponential transmission levels of PCM photonic memory with low quantization error. Given the exponential transmissivity distribution, normal uniform quantization fails to fit it well with severe encoding error. Hence, a dedicated quantizer $q(w, b)$, where b is the bit-width, is required to minimize the quantization error,

$$\min \ \|\hat{w} - w\|_2^2, \quad \text{s.t.} \ \ \hat{w} = q(w, b) \in Q_b, \tag{4}$$

where Q_b denotes a set of quantization levels, i.e., the transmission levels of a b-bit photonic memory cell.

However, as PCM can only demonstrate positive light transmission, to support full-range weight, we store the positive and negative values of weight matrix W in the positive PTC and the negative PTC, respectively. Then, the differential photo-detection module will generate balanced output. With the simple but effective *weight extension strategy*, each scalar weight w in W is expressed as

$$w = w_{pos} - w_{neg}. \tag{5}$$

Here, w_{pos} and w_{neg} are physical values in positive and negative photonic memory cells, where one is selected based on the sign of w to store the weight value and the other is set to the lowest light transmission level δ. In this way, the differential weight encoding in (5) augments the quantization codebook Q_b as follows,

$$Q_b = \{c^{2^b-1} - \delta, \pm(c^{2^b-2} - \delta), \ldots, \pm(c^0 - \delta)\}, \ \delta = c^{2^b-1}, \tag{6}$$

where the number of implementable quantization levels in Q_b is almost doubled. This attributes our quantization with a higher model expressivity, especially under low-bit quantization.

With the augmented quantization codebook, for w within $[-1, 1]$, an augmented base-c quantizer is hence proposed to optimize (4) as

$$w_q = q(w, b) = \frac{\texttt{sign}(w)}{s} \cdot (c^{\texttt{Clip}(\texttt{R}(\log_c(s|w|+\delta)),\ 0,\ 2^b-1)} - \delta), \tag{7}$$

where the scaling factor $s = c^0 - c^{2^b-1}$ is used to transform quantization levels into $[-1, 1]$ and $\texttt{R}(\cdot)$ is a round function. $\texttt{Clip}(\cdot)$ is a clip function to limit the value within $[0, 2^b - 1]$. A quantization-aware training procedure [19] is adopted with our augmented base-c quantizer to train the PCM-based ONNs. In the forward propagation, the weights W are quantized as

$$w_q = q(\frac{\texttt{Tanh}(w)}{\max(\texttt{Tanh}(w))}, b), \ b > 1. \tag{8}$$

In the backward pass, we coarsen the whole b-bit quantization process $q(w, b)$ as an entirety and estimate its gradient g_q by [20]:

$$g_q = \frac{\partial \mathcal{L}}{\partial W} = \frac{\partial \mathcal{L}}{\partial W_q} \frac{\partial W_q}{\partial W} = \frac{\partial \mathcal{L}}{\partial W_q}. \tag{9}$$

Considering limited on-chip storage, a uniform quantizer in [21] is used to discretize layer input.

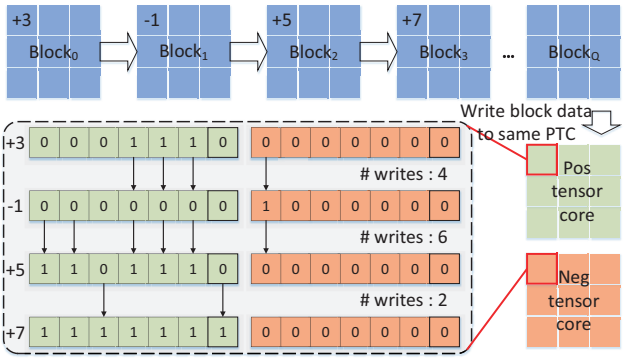


Fig. 3: The illustration of writing multiple weight blocks in the same PTC with redundant wire writes elimination. Each PCM wire within 3-bit memory is in a(1) or c(0) state to demonstrate quantization levels in -7∼+7. Wire-level redundant writes are excluded to reduce *# writes*.

IV. Proposed Aging-Aware Co-Optimization Framework

In this section, we propose the aging-aware co-optimization framework, `ELight`, to minimize both *# total writes* and *# max writes*. We first illustrate the problem formulation and the adopted *augmented redundant write elimination* (ARWE) strategy for wire-level writes in PCM cells. Then we describe the proposed *write-aware training* method to encourage the similarity among weight blocks. At last, we propose a fine-grained *column-based reordering* method further to cut down redundant writes as a post-training optimization strategy.

A. Problem Formulation

Due to the limited scale of a single PTC, we adopt blocking matrix multiplications to implement convolutional layers and fully-connected layers for practical considerations. Specifically, an im2col algorithm [22] converts the convolutions into general matrix multiplication (GEMM). The weight matrix $W \in \mathbb{R}^{M \times N}$ is partitioned into $P \times Q$ sub-matrices, where each $k \times k$ block can be deployed onto one PTC. Then the set B of sub-matrices is assigned to a cluster C of photonic tensor cores. Since we have $M, N \gg k$, the number of sub-matrices is quite large. For example, assuming that the size of PTC is 64×64, B contains 8×72 blocks to implement the 5^{th} convolutional layer of VGG8. Moreover, considering the limited number of on-chip PTCs , multiple sub-matrices need to be assigned to one PTC, leading to massive reuse of PTCs during inference. Here, without loss of generality, we adopt an assignment strategy to assign sub-matrices to PTCs for the following discussion. For one layer with $P \times Q$ blocks, a cluster of PTCs is dedicated for the MM, where one PTC is assigned with a row of weight blocks. P PTCs carry out blocking MM in parallel with shared input. We assume we write new block data into PTC after finishing all the block MMs on the current stored block to reduce reprogramming efforts. It should be noted that our method can work with other assignment schemes.

As data are represented by the binary-state PCM wires within photonic memories, inspired by redundant write elimination (RWE) strategy [23], we propose an augmented redundant write elimination (ARWE) strategy, where we eliminate the writes for identical values and eliminate identical wire-level writes. Concretely, to demonstrate desired value, we preserve the current states of PCM wires at the largest extent by only perturbing the smallest number of wires. For a clear illustration, Fig. 3 shows one example of writing a sequence of weight blocks into 3-bit PTCs. One positive PTC and one negative PTC are used together to demonstrate full-range weights. When

TABLE I: Layer-wise statistics of writes of 5-bit VGG8 model.

	Layer 2	Layer 3	Layer 4	Layer 5
# total writes	1.14×10^6	4.87×10^6	1.66×10^7	3.26×10^7
# max writes	294	534	914	1425

programming transmission level +7 into the memory cell storing level +5, two c-state PCM wires in the positive PTC need to be programmed to a state to achieve the smallest number of writes. Note that the binary state of each PCM wire would be stored in memory using 1bit such that off-chip computers can reprogram based on ARWE strategy, without the need of detecting stored values.

Thus, considering write efforts in both positive and negative PTCs, the number of writes (WT) between two b-bit numbers $w^{'}$ and w is computed as follows,

$$WT(w^{'},w) = |l^{+}(w^{'}) - l^{+}(w)| + |l^{-}(w^{'}) - l^{-}(w)|, \tag{10}$$

where l^{+} and l^{-} denote the transmission level in positive and negative PTC, respectively. The absolute value of l^{+} and l^{-} also indicate the number of a-state wires out of 2^b-1 wires in positive and negative PTCs, respectively, which can be derived based on (7) as

$$l^{+}(w) = \begin{cases} (2^b-1) - \texttt{Clip}(\texttt{R}(\log_t(s|w|+\delta)), 0, 2^b-1), & w \geq 0 \\ 0, & w < 0 \end{cases} \tag{11}$$

$$l^{-}(w) = \begin{cases} 0, & w \geq 0 \\ \texttt{Clip}(\texttt{R}(\log_t(s|w|+\delta)), 0, 2^b-1) - (2^b-1), & w < 0 \end{cases}. \tag{12}$$

Then, we have transmission level of w as $l(w) = l^{+}(w) - l^{-}(w)$, ranging from $-(2^b-1)$ to (2^b-1).

Accordingly, to write new block $A^{'}$ of size $k \times k$ to photonic memories storing block A, the number of writes is computed as

$$WT(A^{'},A) = \sum_{i}^{k}\sum_{j}^{k}(|l^{+}(a_{ij}^{'}) - l^{+}(a_{ij})| + |l^{-}(a_{ij}^{'}) - l^{-}(a_{ij})|). \tag{13}$$

Now we consider the write count of the weight matrix W in the j^{th} layer, which is partitioned into a set B with $P \times Q$ blocks of size $k \times k$. Each PTC t in a cluster C is assigned with n_t blocks, i.e., $\{B_1^t, B_2^t, \ldots, B_{n_t}^t\}$. The number of writes for layer j (LWT) is computed as

$$LWT^j = \sum_{t}^{C}\sum_{i}^{n_t} WT(B_i^t, B_{i-1}^t), \tag{14}$$

where we assume the first block is mapped onto the initialized PTC with all PCM wires being set to c state.

Hence, to deploy a model, the total number of writes is the sum of layer-wise write operations, i.e., $\#total\ writes = \sum_j^L LWT^j$. To evaluate the status of the most over-utilized memory cell, we define a layer-wise metric, *# max writes*, by counting the maximum number of write operations over a single PCM memory cell for a cluster of PTCs. Table. I shows the statistics of the number of writes and the maximal writes for the convolutional layers of 5-bit VGG8. Though our augmented redundant write elimination strategy is applied, we still observe a significant number of writes that challenge the PCM endurance. Therefore, in the following discussion, several techniques are proposed to minimize both *# total writes* and *# max writes*.

B. Write-Aware Training via Block Matching

To reduce the number of write operations when mapping a group of weight blocks, a straightforward technique is to shrink the weight distance between neighboring blocks. Take the example of reducing write operations when writing a weight sequence into one memory cell. As shown in Fig. 4(a), by constraining the distance between neighbors, neighboring weights are encouraged to be similar, thus holding more redundant writes. However, it sets no constraint on the distance between the largest value and the smallest value. This requires more wires to be programmed to demonstrate a wider range. Hence, a wise solution is to constrain weights around a reference value such that the value range is under control, shown in Fig. 4(b).

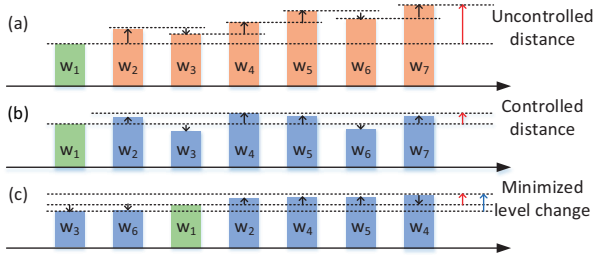


Fig. 4: An example of constraining a weight sequence starting from w_1. (a) Constrain the distance between neighbors. (b) Constrain weights around reference value (w_1). (c) Sort weights in (b) in an ascending order.

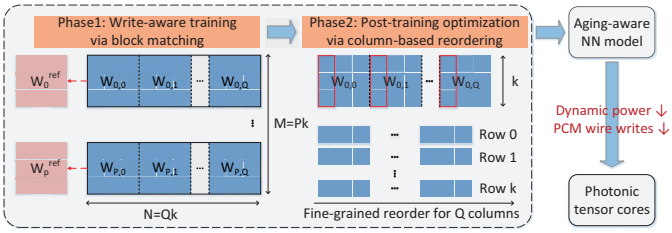


Fig. 5: Proposed two-phase `ELight` to enable efficient photonic in-memory neurocomputing with lifetime enhancement.

To boost the similarity among weight blocks, we propose a write-aware training procedure, shown as phase 1 in Fig. 5. We set the average block as a reference for weight blocks assigned to the same PTC and penalize their transmission level distance from the reference. Instead of directly optimizing the level difference between blocks using (13) in a L1 way, we use an L2 regularization term to calculate the level difference (LD) between block B and A as,

$$LD(B,A) = \sum_{i}^{k}\sum_{j}^{k}\|\tilde{l}^{+}(b_{ij}) - \tilde{l}^{+}(a_{ij})\|^2 + \|\tilde{l}^{-}(b_{ij}) - \tilde{l}^{-}(a_{ij})\|^2. \tag{15}$$

Here, $\tilde{l}^{+}$ and $\tilde{l}^{-}$ denote the normalized l^{+} and l^{-} by dividing $\alpha_b = 2^b - 1$. Normalizing level data to $[-1, 1]$ can help healthy gradient propagation. The L2 regularization term imposes a heavier penalty for large value deviation, and slight deviation is allowed to maintain diverse weights. In this way, not only *# max writes* can be constrained via rejecting large value deviation, but the model expressivity can be mostly maintained. Thus, the block matching loss is defined as

$$\mathcal{L}_{BM} = \sum_{l}^{L}\sum_{t}^{G}\sum_{i}^{n_g}\frac{1}{\beta^B} LD(B_i^t, B_{avr}^t), \tag{16}$$

where we explicitly encourage the similarity of weight blocks in the same group G. β^B is the block size to normalize the level distance.

By adding the block matching loss to the loss function, we trade off between the accuracy and block similarity by controlling λ, as,

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda\mathcal{L}_{BM}, \tag{17}$$

where $\mathcal{L}_{CE}$ is the original cross-entropy loss.

However, there are three issues to optimize $\mathcal{L}_{BM}$. First, it is not differentiable due to the Round$(\cdot)$ operations in (11) and (12). Second, the gradient approximation through the *logarithmic* operations needs to be carefully handled. Third, transmission levels in both positive and negative PTCs are used to compute LD, while only levels that are physically implemented on PTCs need to be involved in gradient evaluation. In other words, only the gradient from either $\|\tilde{l}^{+}(w) - \tilde{l}^{+}(\delta)\|^2$ or $\|\tilde{l}^{-}(w) - \tilde{l}^{-}(\delta)\|^2$ need to be propagated back to compute the gradient w.r.t w depending on its sign. Hence, by leveraging straight-through-estimator (STE) to approximate the gradients for Round$(\cdot)$, we propagate the gradient of $\mathcal{L}_{BM}$ back as,

$$\frac{\partial \mathcal{L}_{BM}}{\partial W} = \frac{1}{\beta B}\Big(\frac{\partial \mathcal{L}_{BM}}{\partial \tilde{l}^{+}(W)}\frac{\mathrm{d}\tilde{l}^{+}(W)}{\mathrm{d}W}\odot\mathcal{M}^{+} + \frac{\partial \mathcal{L}_{BM}}{\partial \tilde{l}^{-}(W)}\frac{\mathrm{d}\tilde{l}^{-}(W)}{\mathrm{d}W}\odot\mathcal{M}^{-}\Big), \quad (18)$$

where $\mathcal{M}^{+}$ and $\mathcal{M}^{-}$ are non-negative and negative masks of W to extract the needed gradients from $\frac{\mathrm{d}\tilde{l}^{+}(W)}{\mathrm{d}W}$ and $\frac{\mathrm{d}\tilde{l}^{-}(W)}{\mathrm{d}W}$, computed by,

$$\frac{\mathrm{d}\tilde{l}^{+}(W)}{\mathrm{d}W}\Big|_{W\geq 0} = \frac{-\mathrm{d}(\log_t(s|W|+\delta))}{\alpha_b \mathrm{d}W} = \frac{-s}{\alpha_b \ln(t)(s|W|+\delta)}, \quad (19)$$

$$\frac{\mathrm{d}\tilde{l}^{-}(W)}{\mathrm{d}W}\Big|_{W<0} = \frac{\mathrm{d}(\log_t(s|W|+\delta))}{\alpha_b \mathrm{d}W} = \frac{-s}{\alpha_b \ln(t)(s|W|+\delta)}. \quad (20)$$

C. Post-Training Optimization via Column-based Reordering

While the technique proposed above boosts the similarity among weight blocks, there is still room for further optimization since it does not consider the mapping order of blocks. We still take Fig. 4(b) as an example. The sum of neighboring differences is not explicitly optimized as we only limit all weights around one reference value. One straightforward method to further reduce the sum of neighboring differences is to sort the weight sequence either in ascending or descending order, as shown in Fig. 4(c).

Inspired by this heuristic, we propose a fine-grained *column-based reordering* method to sort the weights in blocks that share the same PCM memory cells in PTCs. The second phase in Fig. 5 illustrates our idea: For a group of weight blocks assigned to one PTC, weights located in the same position are first shaped into 1-D sequences, then weight sequences are separately sorted in ascending order. We can also sort them in descending order. The weights are finally scattered back to different blocks in the new order. Note that the above process is equivalent to swapping columns element-wise with no influence on final results. Besides, with the aid of the sorting heuristic, when mapping a group of blocks, weight values are written into photonic memories in ascending order. By eliminating redundant writes over wires, *# max writes* over single photonic memory cell can be upper-bounded by the level range of its stored weights, i.e., $2^{b+1}-1$.

Therefore, combining write-aware training with post-training optimization, *# total writes* and *# max writes* can be significantly reduced to mitigate the aging issue and the tedious programming efforts.

V. Experimental Results

To demonstrate the effectiveness of proposed two-phase framework ELight, we conduct experiments on MNIST [24], FashionMNIST [25], CIFAR-10 and CIFAR-100 [26] datasets. On the first two tasks, a CNN configuration C32K4-C32K4-P5-F64-F10 is adopted, where C32K4 is a 4×4 convolutional layer with 32 kernels, P5 means average pooling with output size 5×5, and F64 is a fully-connected (FC) layer with 64 neurons. On CIFAR-10 and CIFAR-100, VGG8 [27] and VGG13 [28] are used, while we replace the last three FC layers with one FC layer to avoid over-fitting. We implement all models in PyTorch on a machine with an Intel Core i7-9700 CPU and an NVIDIA Quadro RTX 6000 GPU. The CNN and

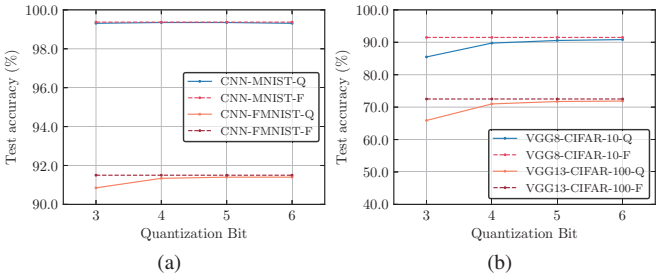


Fig. 6: Quantization evaluation on (a) CNN and (b) VGG models. F means full-precision models. Q means models with quantization.

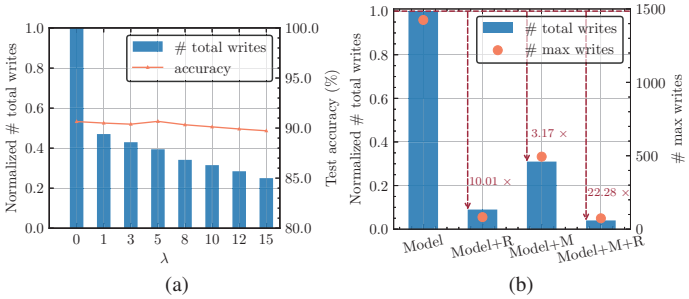


Fig. 7: Evaluation of proposed aging-aware optimization techniques on 5-bit VGG8. (a) Normalized *# total writes* and accuracy comparison with different λ for write-aware training. (b) Comparison between *# total writes* and *# max writes* of the 5^{th} convolutional layer.

VGG models are trained for 100 and 200 epochs, respectively, using the SGD optimizer with a momentum of 0.9. Regarding the photonic tensor core size, we assume 16×16 and 64×64 for the small CNN and VGG model, respectively. Note that for fair comparison, the *augmented redundant write elimination* (ARWE) strategy is also applied in the baseline to collect statistics.

A. Evaluation of the Distribution-Aware Quantization Scheme

In Fig. 6, we evaluate our augmented base-c quantizer under 3- to 6-bit quantization. Our quantization scheme can successfully tackle the issue of non-linear transmission level distribution, enlarges the solution space, and achieves high accuracy under low-bit quantization. Under 4- to 6-bit quantization, our proposed method can achieve negligible or small accuracy loss on all tasks. Our method can still maintain $> 85\%$ on CIFAR-10 and $> 65\%$ accuracy on CIFAR-100 under 3-bit quantization for relatively complicated tasks.

B. Evaluation of Proposed Aging-Aware Optimization Framework

1) Evaluation of write-aware training via block matching: To investigate the impact of write-aware training, we visualize the normalized *# total writes* and accuracy of 5-bit VGG8 with various degrees of λ in Fig. 7a. With the increase of λ, *# total writes* decreases since of the existence of stronger similarity among blocks. With too large λ, accuracy starts to decline since weights cannot be identical even we pursue block-level similarity. Otherwise, it will be hard for weights to capture diverse features. By trading off the accuracy and *# total writes*, a sweet point ($\lambda = 10$) exists with $3.17\times$ reduction in *# total writes* and only 0.44% accuracy drop. Interestingly, sometimes a proper λ leads to better accuracy as a regularization mechanism.

2) Evaluation of post-training optimization via column-based reordering: Fig. 7b shows the comparison of *# total writes* of 5-bit VGG8 between (1) Model trained with/without column-based

TABLE II: Performance of ELight on VGG networks on CIFAR-10 and CIFAR-100 dataset. AC: accuracy change, R: column-based reordering. The *# max writes* of one largest convolutional layer (the 5^{th} convolutional layer for VGG8 and 8^{th} convolutional layer for VGG13) is shown here.

Network	Dataset	Bitwidth	λ	Acc(%)/AC	*# total writes* ↓ (×)		Energy cost ↓ (×)		*# max writes*	
					-	+R	-	+R	-	+R
VGG8	CIFAR-10	3	0	86.71	1	6.52	1	9.27	128	15
			8	86.02/-0.69	22.12	**46.11**	6.63	**69.29**	14	**7**
		4	0	89.75	1	7.84	1	11.31	401	36
			10	89.94/+0.19	3.83	**24.45**	3.92	**35.48**	95	**19**
		5	0	90.56	1	10.01	1	14.35	1425	82
			10	90.12/-0.44	3.17	**22.28**	3.20	**31.17**	494	**74**
		6	0	90.83	1	12.31	1	16.89	4464	180
			5	89.88/-0.95	6.82	**26.35**	7.15	**32.48**	1560	**146**
VGG13	CIFAR-100	4	0	70.99	1	9.66	1	13.84	542	39
			10	70.44/-0.55	3.54	**29.25**	3.57	**42.02**	173	**33**
		5	0	71.73	1	12.06	1	17.29	1771	84
			3	71.95/+0.22	2.19	**21.93**	2.21	**31.41**	921	**55**
		6	0	71.88	1	14.37	1	17.62	4926	182
			3	70.97/-0.91	3.11	**22.65**	3.19	**29.85**	3577	**156**

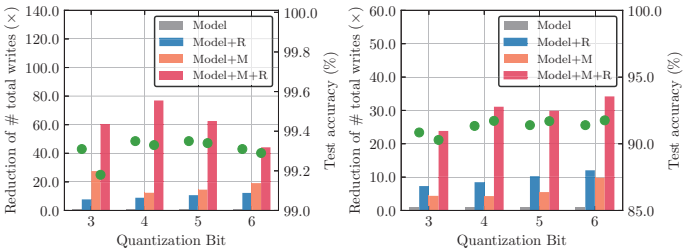


Fig. 8: The reduction of # total writes and accuracy of CNN model trained on (a) MNIST and (b) FashionMNIST under different bit-width.

reordering (R), (2) Model trained with block matching loss (M) with/without column-based reordering (R). We also compare the *# max writes* of mapping the 5-th convolutional layer. The results provide three insights: (1) The column-based reordering can reduce *# total writes* a lot even without extra training efforts to boost weight similarity, achieving $10.01\times$ reduction. (2) With the write-aware training to boost similarity among blocks, the column-based reordering can achieve the best of reduction on *# total writes* by $22.28\times$, which justifies the effectiveness of our joint aging-aware optimization framework. (3) Our proposed column-based reordering upper-bounds *# max writes* by the number of transmission levels, i.e., $2^{b+1}-1$, which eliminates any redundant writes and ensures that only a small number of PCM wires need to be re-written.

3) Evaluation on the synergistic optimization framework: To further testify the effectiveness of the proposed aging-aware optimization framework, we evaluate different models under different bit-width quantization. For practical use, we choose a sweet point of λ to constrain the accuracy drop within 1%. Fig. 8 shows the reduction of *# total writes* and accuracy of the CNN model on MNIST and FashionMNIST under 3- to 6-bit quantization. On those simple tasks, a significant reduction of *# total writes* is obtained with negligible accuracy drop. Table. II shows the effectiveness of our proposed techniques on VGG8 and VGG13 trained on CIFAR-10 and CIFAR-100, respectively. The proposed write-aware training and post-training optimization techniques can work orthogonally to each other, achieving the largest reduction on *# total writes* and *# max writes*, where *# total writes* can be reduced by $> 20\times$ with less than 1% accuracy degradation. Hence, our joint optimization framework can successfully mitigate the aging issue by largely reducing the number of write operations.

TABLE III: Pulse profiles for $a \rightarrow c$ and $c \rightarrow a$ transition [12].

	Pulse period(μs)	Pulse voltage(V)	# Pulse
$a \rightarrow c$	1	5	20
$c \rightarrow a$	0.5	15	1

4) Evaluation of power saving: To testify the energy efficiency when applying the above optimization methods, we trace the detailed energy cost of writing weight block data onto PTCs during the inference process. Assuming the resistance of heaters is consistent, the ratio of write energy cost between a-c and c-a transition is $40:9$ using the pulse profiles on external heaters in Table III. Thanks to the reduction of *# total writes*, the energy cost of deploying VGG8 and VGG13 is reduced by $> 30\times$ under different bit-widths, as shown in Table. II, which proves our aging-aware optimization framework effectively saves dynamic energy cost incurred by PTC reuse.

VI. Conclusion

In this work, we propose a synergistic solution to enable efficient photonic in-memory neurocomputing with an enhanced lifetime. First, we model the non-linear transmission distribution of PCM-based photonic memories and propose a distribution-aware quantization scheme to reduce weight encoding errors. Based on this, we propose a synergistic aging-aware optimization framework `ELight` to trim down redundant PCM writes via block-matching-based training and column-based reordering. Our co-optimization method significantly reduces the number of total write operations and the number of write operations for the most over-utilized memory cell. Experimental results demonstrate that the proposed optimization framework can reduce the number of write operations and energy costs by $>20\times$, pushing photonic in-memory neurocomputing towards practical application in efficient machine learning.

Acknowledgement

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] Y. Shen, N.C. Harris, S. Skirlo *et al.*, "Deep learning with coherent nanophotonic circuits," *Nature Photonics*, 2017.
[2] Z. Zhao, D. Liu, M. Li *et al.*, "Hardware-software co-design of slimmed optical neural networks," in *Proc. ASPDAC*, 2019.
[3] J. Gu, Z. Zhao, C. Feng *et al.*, "Towards area-efficient optical neural networks: an FFT-based architecture," in *Proc. ASPDAC*, 2020.

[4] W. Liu, W. Liu, Y. Ye *et al.*, "Holylight: A nanophotonic accelerator for deep learning in data centers," in *Proc. DATE*, 2019.
[5] Q. Cheng, J. Kwon, M. Glick *et al.*, "Silicon Photonics Codesign for Deep Learning," *Proceedings of the IEEE*, 2020.
[6] F. Zokaee, Q. Lou, N. Youngblood *et al.*, "LightBulb: A Photonic-Nonvolatile-Memory-based Accelerator for Binarized Convolutional Neural Networks," in *Proc. DATE*, 2020.
[7] G. Wetzstein, A. Ozcan, S. Gigan *et al.*, "Inference in artificial intelligence with deep optics and photonics," *Nature*, 2020.
[8] J. Gu, Z. Zhao, C. Feng *et al.*, "O2NN: Optical Neural Networks with Differential Detection-Enabled Optical Operands," in *Proc. DATE*, Feb. 2021.
[9] J. Gu, C. Feng, Z. Zhao *et al.*, "SqueezeLight: Towards Scalable Optical Neural Networks with Multi-Operand Ring Resonators," in *Proc. DATE*, Feb. 2021.
[10] B.J. Shastri, A.N. Tait, T.F. de Lima *et al.*, "Photonics for artificial intelligence and neuromorphic computing," *Nature Photonics*, 2021.
[11] J. Gu, Z. Zhao, C. Feng *et al.*, "Towards Hardware-Efficient Optical Neural Networks: Beyond FFT Architecture via Joint Learnability," *IEEE TCAD*, 2020.
[12] M. Miscuglio and V.J. Sorger, "Photonic tensor cores for machine learning," *Applied Physics Review*, 2020.
[13] J. Feldmann, N. Youngblood, M. Karpov *et al.*, "Parallel convolution processing using an integrated photonic tensor core," *Nature*, 2021.
[14] Y. Zhang, C. Ríos, M.Y. Shalaginov *et al.*, "Myths and truths about optical phase change materials: A perspective," *Applied Physics Letters*, 2021.
[15] Y. Cai, Y. Lin, L. Xia *et al.*, "Long live time: improving lifetime for training-in-memory engines by structured gradient sparsification," in *Proc. DAC*, 2018.
[16] M. Liu, L. Xia, Y. Wang *et al.*, "Fault tolerance in neuromorphic computing systems," in *Proc. ASPDAC*, 2019.
[17] W. Wen, Y. Zhang, and J. Yang, "Renew: Enhancing lifetime for reram crossbar based neural network accelerators," in *Proc. ICCD*, 2019.
[18] F. Meng, Y. Xue, and C. Yang, "Power-and endurance-aware neural network training in nvm-based platforms," *IEEE TCAD*, 2018.
[19] S. Zhou, Z. Ni, X. Zhou *et al.*, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *Arxiv*, 2016.
[20] G. Hinton, "Neural networks for machine learning," *Coursera Video Lecture*, 2012.
[21] A. Fan, P. Stock, B. Graham *et al.*, "Training with quantization noise for extreme model compression," in *Proc. ICML*, 2021.
[22] Y. Jia, E. Shelhamer, J. Donahue *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014.
[23] B.D. Yang, J.E. Lee, J.S. Kim *et al.*, "A low power phase-change random access memory using a data-comparison write scheme," in *Proc. ISCAS*, 2007.
[24] Y. LeCun, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/, 1998.
[25] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms," *Arxiv*, 2017.
[26] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.
[27] L. Deng, P. Jiao *et al.*, "Gxnor-net: Training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework," *Neural Networks*, 2018.
[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Arxiv*, 2014.