Real-Time FJ/MAC PDE Solvers via Tensorized, Back-Propagation-Free Optical PINN Training

Yequan Zhao[1, *], **Xian Xiao**[2, *], **Xinling Yu**[1], **Ziyue Liu**[1], **Zhixiong Chen**[1],
Geza Kurczveil[2], **Raymond G. Beausoleil**[2], **Zheng Zhang**[1]
[1] University of California, Santa Barbara
[2] Hewlett Packard Labs, Hewlett Packard Enterprise
[*] Equal Contributions

Abstract

Solving partial differential equations (PDEs) numerically often requires huge computing time, energy cost, and hardware resources in practical applications. This has limited their applications in many scenarios (e.g., autonomous systems, supersonic flows) that have a limited energy budget and require near real-time response. Leveraging optical computing, this paper develops an on-chip training framework for physics-informed neural networks (PINNs), aiming to solve high-dimensional PDEs with fJ/MAC photonic power consumption and ultra-low latency. Despite the ultra-high speed of optical neural networks, training a PINN on an optical chip is hard due to (1) the large size of photonic devices, and (2) the lack of scalable optical memory devices to store the intermediate results of back-propagation (BP). To enable realistic optical PINN training, this paper presents a scalable method to avoid the BP process. We also employ a tensor-compressed approach to improve the convergence and scalability of our optical PINN training. This training framework is designed with tensorized optical neural networks (TONN) for scalable inference acceleration and MZI phase-domain tuning for *in-situ* optimization. Our simulation results of a 20-dim HJB PDE show that our photonic accelerator can reduce the number of MZIs by a factor of 1.17×10^3, with only 1.36 J and 1.15 s to solve this equation. This is the first real-size optical PINN training framework that can be applied to solve high-dimensional PDEs.

1 Introduction

Partial differential equations (PDEs) are used to describe numerous science and engineering problems. In practical engineering design, solving a PDE via discretization-based numerical methods (e.g., finite difference or finite elment) normally requires a huge amount of computing resources and run-time due to the resulting large-scale algebraic equations. As a result, traditional PDE solvers are often run on a powerful workstation or HPC platform. Recently, physics-informed neural networks (PINN)[1, 2, 3] have emerged as a promising meshless approach to solve high-dimensional or parametric PDEs in both forward and inverse problems.

While PINN can overcome the curse of dimensionality caused by numerical discretizations, training a realistic PINN is still expensive in many cases, limiting their applications in real-time scenarios where repeated and fast training is required. For instance, in safety verification and control of autonomous systems, a Hamiltonian-Jacobi-Issac (HJI) PDE or a Hamiltonian-Jacobi-Bellman (HJB) PDE has to be solved repeatedly as the sensor data and avoidance specification updates. Training such a PINN on a powerful GPU can take over 20 hours [4, 5], whereas there are strict requirements on the latency and energy cost of the embedded computing platforms. This prevents the real-time safety-aware decision making for autonomous systems. In medical imaging such as electrical property tomography [6],

37th First Workshop on Machine Learning with New Compute Paradigms at NeurIPS 2023(MLNPCP 2023).

each training can take dozens of hours, and the measured MRI data is private. It is highly desirable to speed up the training on a local edge device. This motivates the training of PINNs on light edge devices to enable real-time sensing and decision making.

Optical neural network (ONN) accelerators provide a promising solution for real-time inference and training [7, 8, 9]. However, training PINNs on photonic chips is very challenging due to three constraints. Firstly, photonic multiply-accumulate (MAC) units such as Mach-Zehnder interferometers (MZIs) are much larger ($\sim$10s of microns) than CMOS transistors, resulting in a low integration density. A real-size PINN with $> 10^5$ model parameters can easily exceed the available chip size with the square scaling rule where an $N \times N$ optical weight matrix requires $O(N^2)$ MZIs [10, 11]. Secondly, it is hard to realize on-chip training on photonic chips. Several back-propagation(BP)-free methods are proposed to circumvent the "hardware-unfriendly" nature of error feedback in BP [12, 13, 14, 15, 16]. Unfortunately, these methods are also limited by their scalability issue. Thirdly, the loss for PINN training includes higher-order derivatives that require multiple BPs to accurately compute. Due to the inefficiency of *in-situ* BP [17, 18], an alternative numerical method is needed for photonic implementation.

Paper Contributions. This paper proposes the first optical training framework that can handle realistic large-size PINNs on the integrated photonic platform. Our major contributions include:

- We employ a BP-free approach using only additional inferences to calculate gradient and derivative estimation, enabling training PINN and solving realistic PDEs on a photonic chip.
- We utilize a tensor-compressed format to reduce the number of photonic devices and to improve the convergence of the BP-free optical PINN training framework.
- We demonstrate numerical simulation of the optical PINN training method to solve a 20-dimensional HJB PDE. Our method is robust to hardware imperfection and achieves competitive performance while reducing 1.17×10^3 MZI devices and requiring only 1.36 J and 1.15 s to solve this PDE.

Our approach greatly advances the state-of-the-art, and it can handle optical training of fully connected networks with sizes up to 1024×1024. This work will pave the way for future real-time and fJ/MAC computing for solving complex high-dimensional PDEs.

2 Preliminaries

2.1 Optical Neural Networks (ONN) and Tensorized Optical Neural Networks (TONN)

We focus on the ONN [7] architecture with singular value decomposition (SVD) to implement matrix-vector multiplication (MVM), i.e., $y = \boldsymbol{W}x = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{*}x$. The unitary matrices $\boldsymbol{U}$ and $\boldsymbol{V}^{*}$ are implemented by MZIs in Clements mesh [11]. The parametrization of $\boldsymbol{U}$ and $\boldsymbol{V}^{*}$ is given by $\boldsymbol{U}(n) = \boldsymbol{D}\prod_{i=2}^{n}\prod_{j=1}^{i-1}\boldsymbol{R}_{ij}\left(\phi_{ij}\right)$ where $\boldsymbol{D}$ is a diagonal matrix, and each 2-dimensional rotator $\boldsymbol{R}_{ij}(\phi_{ij})$ can be implemented by a 2×2 MZI containing two phase shifters and two 50/50 splitters. We denoted all programmable phases as $\boldsymbol{\Phi}$ and $\boldsymbol{W}$ is parametrized as $\boldsymbol{W}(\boldsymbol{\Phi})$.

To increase the scalability of ONN, a tensorized optical neural network (TONN)[19] is proposed to realize large-scale ONNs with reduced hardware resources (i.e., MZIs) using the tensor-train (TT) decomposition algorithm. Let $\boldsymbol{W} \in \mathbb{R}^{M\times N}$ be a generic weight matrix in a neural network. We factorize its dimension sizes as $M = \prod_{i=1}^{L} m_i$ and $N = \prod_{j=1}^{L} n_j$, fold $\boldsymbol{W}$ into a $2L$-way tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{m_1\times m_2\times\cdots\times m_L\times n_1\times n_2\times\cdots\times n_L}$, and parameterize $\boldsymbol{\mathcal{W}}$ with the TT decomposition [20]:

$$\boldsymbol{\mathcal{W}}(i_1, i_2, \ldots, i_L, j_1, j_2, \ldots, j_L) \approx \prod_{k=1}^{L} \mathbf{G}_k(i_k, j_k) \tag{1}$$

Here $\mathbf{G}_k(i_k, j_k) \in \mathbb{R}^{r_{k-1}\times r_k}$ is the (i_k, j_k)-th slice of the TT-core $\boldsymbol{\mathcal{G}}_k \in \mathbb{R}^{r_{k-1}\times m_k\times n_k\times r_k}$by fixing its 2nd index as i_k and 3rd index as j_k. The vector $(r_0, r_1, \ldots, r_L)$ is called TT-ranks with the constraint $r_0 = r_L = 1$. This TT representation reduces the number of unknown variables from $\prod_{k=1}^{L} m_k n_k$ to $\sum_{k=1}^{L} r_{k-1} m_k n_k r_k$. The detailed architecture of TONN can be found in [19].

2.2 Physics-Informed Neural Networks (PINNs) and Tensor-compressed PINNs

Consider the well-posed initial value partial differential equation (PDE) problem described by:

$$\begin{aligned} \mathcal{N}[\boldsymbol{u}(\boldsymbol{x}, t)] &= l(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega,\ \ t \in [0, T], \\ \mathcal{I}[\boldsymbol{u}(\boldsymbol{x}, 0)] &= g(\boldsymbol{x}), \quad\ \ \boldsymbol{x} \in \Omega, \end{aligned} \tag{2}$$

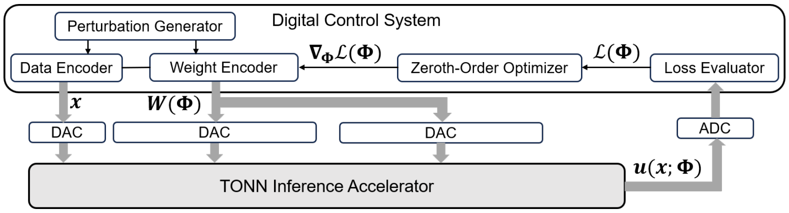

Figure 1: The overall architecture of the BP-free optical training accelerator.

where $\boldsymbol{x}$ and t are the spatial and temporal coordinates; $\Omega \subset \mathbb{R}^D$ and T denote the spatial domain and time horizon, respectively; $\mathcal{N}$ is a general nonlinear differential operator; $\mathcal{I}$ represents the initial condition; $\boldsymbol{u} \in \mathbb{R}^n$ is the solution for the PDE described above. In PINNs [3], a neural network $\boldsymbol{u}(\boldsymbol{x}, t; \boldsymbol{\theta})$, parameterized by $\boldsymbol{\theta}$, is substituted into PDE (2), resulting in a residual defined as:

$$r(\boldsymbol{x}, t; \boldsymbol{\theta}) := \mathcal{N}[\boldsymbol{u}(\boldsymbol{x}, t; \boldsymbol{\theta})] - l(\boldsymbol{x}, t). \tag{3}$$

The parameters $\boldsymbol{\theta}$ can be obtained by minimizing the loss $\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_r(\boldsymbol{\theta}) + \lambda \mathcal{L}_0(\boldsymbol{\theta})$, where

$$\mathcal{L}_r(\boldsymbol{\theta}) = \frac{1}{N_r} \sum_{i=1}^{N_r} \left\| r(\boldsymbol{x}_{\boldsymbol{r}}^i, t_r^i; \boldsymbol{\theta}) \right\|_2^2 \text{ and } \mathcal{L}_0(\boldsymbol{\theta}) = \frac{1}{N_0} \sum_{i=1}^{N_0} \left\| \mathcal{I}[\boldsymbol{u}(\boldsymbol{x}_{\boldsymbol{0}}^i, 0; \boldsymbol{\theta})] - g(\boldsymbol{x}_{\boldsymbol{0}}^i) \right\|_2^2, \tag{4}$$

are the residuals of the PDE and the initial (or terminal) condition, respectively. To adapt PINNs to the edge with constraints in memory, computation, and energy, [21] introduced a tensor-compressed PINN framework, where fully-connected layers are decomposed into a series of TT-cores, as in (1).

3 BP-Free Optical Training Accelerator for Tensor-Compressed PINN

3.1 Overall Architecture

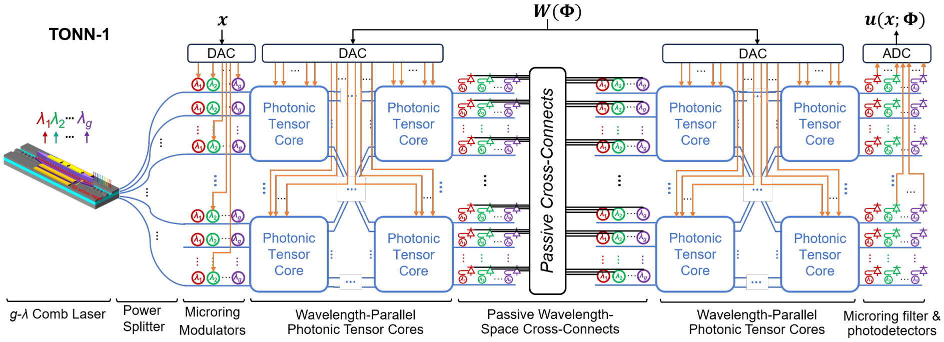

Figure 2: TONN-1: The designed tensor-compressed optical inference accelerator based on the TONN architecture with wavelength and space multiplexing.

The block diagram of our optical PINN training accelerator is shown in Fig. 1. This accelerator does not perform any BP. Instead, it repeatedly call an optical inference accelerator TONN[19] to obtain some loss information. The collected loss information is process in a digital control system, and gradient information is estimated via a zeroth-order optimization to update PINN model parameters. Since propagations are not used, intermediate results will not be computed or stored on the photonic chip.

In the following, we give the details of our TONN design and BP-free training method.

3.2 Tensor-compressed Optical Inference Accelerator Design

We present two designs of optical neural networks based on tensor-train (TT) compression. The TT-based optical neural network design can greatly reduce the number of photonic devices, latency and energy cost. Furthermore, it can reduce the number of on-chip training variables and improve the convergence of the on-chip training framework.

The first design, called TONN-1, is illustrated in Fig. 2. In this design, the tensor multiplications between the input data and all tensor-train cores (the whole tensorized matrix) are realized in a single clock cycle by cascading the photonic tensor cores in the space domain and adding parallelism in the wavelength domain[19].

The second design, called TONN-2 and shown in Fig. 3, uses a single wavelength-parallel photonic tensor core [22] with time multiplexing. Compared with TONN-1, TONN-2 exhibits a smaller footprint at the expense of higher latency and additional memory requirements. In each clock cycle, the photonic tensor core with parallel processing in the wavelength domain is updated to multiply with the input tensor. Then, the intermediate output data is stored in the buffer for the next cycle.

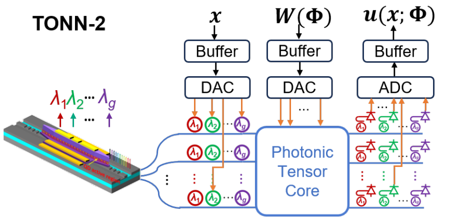
TONN-2
$\boldsymbol{x}$
$\boldsymbol{W}(\boldsymbol{\Phi})$
$\boldsymbol{u}(\boldsymbol{x};\boldsymbol{\Phi})$
Buffer
Buffer
Buffer
DAC
DAC
ADC
$\lambda_1\lambda_2\cdots\lambda_g$
Photonic Tensor Core

Figure 3: TONN-2: The designed inference accelerator using a single wavelength-parallel photonic tensor core with time multiplexing.

3.3 Tensor-Compressed BP-Free PINNs Training

We adopt the idea of [23] to implement a fully BP-free PINN training method to mitigate the memory bottleneck of on-chip photonic computing. In PINN training, the BP process should be avoided in both the loss function evaluation and in the SGD-type optimization step.

BP-free Loss Evaluation. The differential operator in (2) involves first-order and high-order derivatives of $\boldsymbol{u}$ with respect to $\boldsymbol{x}$. It is hard to compute these derivatives via a BP process on a photonic chip. Two methods can be used to address this issue. The first method is finite difference, which calculates the derivatives by perturbing each element of $\boldsymbol{x}$. An alternative method uses sparse-grid Stein estimator [23]. Both method only require a few additional inferences with coordinate-wise perturbed input data to estimate first- and second-order derivatives, then compute $\mathcal{L}(\boldsymbol{\Phi})$. MZIs do not need to be re-programmed when estimating the derivatives.

BP-free Gradient Estimation in SGD-type Optimizers. Stochastic gradient descent (SGD) and its variants are the mainstream optimizer for neural network training. In our optical PINN training framework, we use a zeroth-order gradient estimator, Simultaneous Perturbation Stochastic Approximation (SPSA) [24] to obtain a randomized estimation of the gradient. Specifically, given a model parameterized by $\boldsymbol{\Phi} \in \mathbb{R}^d$ and a loss function $\mathcal{L}$, SPSA computes a randomized gradient estimation

$$\hat{\nabla}_{\boldsymbol{\Phi}}\mathcal{L}(\boldsymbol{\Phi}) = \sum\nolimits_{i=1}^{N} \frac{1}{N\mu} \left[\mathcal{L}\left(\boldsymbol{\Phi} + \mu\boldsymbol{\xi}_i\right) - \mathcal{L}(\boldsymbol{\Phi})\right] \boldsymbol{\xi}_i. \tag{5}$$

Here $\{\boldsymbol{\xi}_i \in \mathbb{R}^d\}_{i=1}^N$, are N i.i.d. samples drawn from $\mathcal{N}(0, \boldsymbol{I}_d)$ and μ is the sampling radius. In practice, we further adopt the concept from signSGD [25] and its ZO counterpart, ZO-signSGD [26], to de-noise the SPSA gradient estimation by preserving only the sign for each update. Specifically, given a learning rate α, the PINN model parameters are updated as

$$\boldsymbol{\Phi}_t \leftarrow \boldsymbol{\Phi}_{t-1} - \alpha \text{sign}(\hat{\nabla}_{\boldsymbol{\Phi}}\mathcal{L}(\boldsymbol{\Phi})) \tag{6}$$

Note that we fully leverage the benefits of the tensor-compressed model in both inference and training. The neural network $\boldsymbol{u}_{\boldsymbol{\Phi}}(\boldsymbol{x}, t)$ is parameterized by all programmable MZI phases $\boldsymbol{\Phi}$ in each photonic TT-core $\boldsymbol{\mathcal{G}}_k(\boldsymbol{\Phi}_k)$. The photonic TT-cores that approximate a weight matrix are directly employed in the inference and updated in the training. Since the gradient variance of the SPSA method grows as the dimensionality of training variables increase, the tensor-compressed format can dramatically reduce the gradient estimation variance and improve the convergence of the ZO training framework.

SPSA requires N additional loss evaluations to estimate the gradients. During the training process, after evaluating $\mathcal{L}(\boldsymbol{\Phi})$, the digital control system generates a perturbation vector and program all MZIs simultaneously. Then, the same training data is shed into the inference accelerator again to conduct the additional inferences $\mathcal{L}(\boldsymbol{\Phi} + \boldsymbol{\mu}\boldsymbol{\xi}_i)$. After N additional inferences, the digital control system averages over the N loss values and then estimates the gradient $\hat{\nabla}_{\boldsymbol{\Phi}}\mathcal{L}(\boldsymbol{\Phi})$, finally updates all MZIs with their updated value simultaneously.

4 Experiments Results

We evaluate our proposed BP-free tensor-compressed PINN training by training a PINN arising from high-dim optimal control of robots and autonomous systems. We consider the following 20-dim HJB PDE:

$$\begin{aligned} &\partial_t u(\boldsymbol{x},t) + \Delta u(\boldsymbol{x},t) - 0.05 \left\| \nabla_{\boldsymbol{x}} u(\boldsymbol{x},t) \right\|_2^2 = -2, \\ &u(\boldsymbol{x},1) = \left\| \boldsymbol{x} \right\|_1, \quad \boldsymbol{x} \in [0,1]^{20}, \ t \in [0,1]. \end{aligned} \tag{7}$$

Here $\left\| \cdot \right\|_p$ denotes an ℓ_p norm. The exact solution is $u(\boldsymbol{x},t) = \left\| \boldsymbol{x} \right\|_1 + 1 - t$. The baseline neural network is a 3-layer optical neural network ($21 \times n, n \times n, n \times 1$, n denotes the number of neurons in the hidden layer) with sine activation. We approximate the solution by a transformed neural network $u(\boldsymbol{x}, t; \boldsymbol{\Phi}) = (1-t) f(\boldsymbol{x}, t; \boldsymbol{\Phi}) + \left\| \boldsymbol{x} \right\|_1$, where $f(\boldsymbol{x}, t; \boldsymbol{\Phi})$ is the base neural network or its TT-compressed version. We remark that the transformed network is designed to ensure our approximated solution exactly satisfies the terminal condition.

4.1 Numerical Simulation Results

All numerical simulations are based on a software implementation built upon PyTorch [27] backend and TorchONN library [28] to simulate the computational model of an optical computing platform. To show the effectiveness and robustness of our design, we compared our method with different training paradigms. **Off-chip Training** denotes first pre-training on electrical digital platforms, e.g., CPUs and GPUs, then mapping the trained model to photonic devices. The gradients w.r.t. model parameters and the derivatives w.r.t. the input are computed by BP. Our proposed tensor-compressed BP-free training belongs to **On-chip Training** as it directly tunes photonic devices (i.e., phase-shifters in MZIs) on-chip and trains from scratch. For off-chip training, we implemented hardware-aware training that incorporates various hardware imperfections and its counterpart hardware-unaware training that runs on an ideal computational model. The hardware-aware training is a hardware-restricted learning problem, where we considered phase-shifter γ coefficient drift $\boldsymbol{\Gamma} \sim \mathcal{N}(\gamma, \sigma_\gamma^2)$ [13, 29] caused by fabrication variations and thermal cross-talk between adjacent devices $\boldsymbol{\Omega}$ [13, 29, 30], and phase bias due to manufacturing error $\boldsymbol{\Phi}_b \sim \mathcal{U}(0, 2\pi)$ and the objective became $\boldsymbol{\Phi}^* = \arg\min_{\boldsymbol{\Phi}} \mathcal{L}(\boldsymbol{W}(\boldsymbol{\Omega}\boldsymbol{\Gamma}\boldsymbol{\Phi} + \boldsymbol{\Phi}_b))$. For on-chip training, we incorporate the same hardware imperfections to mimic the actual analog hardware.

Table 1: Software simulation results. Both ONN and TONN are three-layer MLPs with sine activation. Off. denotes off-chip training, On. denotes on-chip training, w/o and w/ noise denote hardware-unaware and -aware training, respectively. For off-chip training, we reported the validation loss after mapping to hardware with noise and the original validation loss (in parentheses). For on-chip training, we reported the final validation loss.

Network	Neurons	Params	Off. w/o noise	Off. w/ noise	On. w/ noise (proposed)
ONN	1024	608,257	3.10E-01 (7.63E-03)	3.07E-01 (7.81E-03)	1.43E-02
TONN	1024	1,536	3.73E-01 (1.46E-02)	2.97E-01 (1.35E-02)	**5.53E-03**

Our results are provided in Table:1. We report the validation loss which is the mean square error (MSE) w.r.t. the ground truth. After training, our proposed BP-free tensor-compressed PINN training achieves a validation loss of 5.53E-3, indicating that the model fits the ground truth well. The tensor-compressed ONN outperforms the un-compressed ONN, indicating that our tensor-compressed training is capable of preserving the expressive power of a wide ONN with greatly reduced model parameters (396× fewer in this case).

Off-chip training achieves a similar validation loss on the pre-trained model. However, after mapping to real photonic devices, the performance greatly degrades due to the hardware imperfection. The hardware-aware training does not help significantly as the imperfection model in software is not identical to real hardware. Our proposed method inherently circumvents this problem as it directly tunes on the fabricated hardware during on-chip training, thus demonstrating better robustness and better performance.

4.2 System Performance

The system performance for the accelerators based on ONNs and TONNs are evaluated and compared, as shown in Table:2, assuming the III-V-on-Si device platform [31]. Since we only use several additional inferences to estimate the gradients and derivatives, a multiplication of number of inference and energy consumption or latency per inference indicates the energy and training efficiency, respectively. For the TONN design, the first two MLP layers are both factorized as 1024*1024=[4*8*4*8]*[8*4*8*4] with TT-ranks as [1,2,1,2,1]. The total number of wavelengths used is 32 [19]. The SVD implementation of the arbitrary matrices is considered in the calculation.

Table 2: Comparison of the # of MZIs, energy/inference, latency, and photonic footprint

Network	Params	# of MZIs	Energy /inference (J)	Latency /inference (ns)	Footprint (mm^2)
ONN	6.08E05	2.10E06	-	600	2.62E05
TONN-1	1.53E03	1.79E03	6.45E-09	550	648
TONN-2	1.53E03	28	5.05E-09	3604	26

Energy Consumption: The total energy of the accelerators is mainly consumed in the ADC, DAC, and digital control systems. Here, we focus on the photonic energy consumption per forward, which consists of five parts: laser wall-plug power, microring modulator power, MZI mesh power, microring add-drop filter power, and PD receiver power. The conventional ONN has insurmountable optical loss due to the square scaling rule, so the energy cannot be calculated. The TONN-2 consumes slightly less energy per forward due to lower insertion loss even though it requires 64 cycles.

Latency: The latency per inference in on-chip training is calculated by: $t_{\text{inference}} = n_{\text{cycle}} * (t_{\text{DAC}} + t_{\text{tuning}} + t_{\text{opt}} + t_{\text{ADC}}) + t_{\text{DIG}}$, where t_{DAC} is the ADC conversion delay (∼24 ns), t_{tuning} is the metal-oxide-semiconductor capacitor (MOSCAP) phase shifter tuning delay (∼0.1 ns), t_{opt} is the propagation latency of optical signal (∼51.2 ns for ONN, ∼1.6 ns for TONN-1, and ∼0.4 ns for TONN-2), t_{ADC} is the DAC delay(∼24 ns), and t_{DIG} is the digital computation overhead (∼500 ns) for gradient calculation and phase updates. The TONN-2 uses 64 cycles for one inference, while ONN and TONN-1 only needs one cycle.

Training Efficiency: In our 20D-HJB example, we need 42 inferences for each loss evaluation and 10 loss evaluations for gradient estimation. Suppose a mini-batch size of 100, 4.20E4 inferences are required for one epoch. The energy consumption per epoch is estimated as 2.71E-04 J and the latency per epoch is estimated as 0.23 ms for TONN-1. On average training reaches a good solution after 5000 epochs, which corresponds to 1.36 J and 1.15 s for solving a 20D-HJB equation.

Footprint: Only the footprint of the photonic devices, which occupy the major area of the accelerator, is used for comparison. The photonic footprint includes the areas of hybrid silicon comb laser, microring resonator (MRR) modulator arrays, photonic tensor cores, MRR add-drop filters, photodiodes, and electrical cross-connects. It can be seen that TONN-2 occupies a much smaller footprint than TONN-1 at the expense of much higher computational latency.

5 Conclusion

In this work, we have proposed the first optical training framework that can handle realistic large-size PINNs on the integrated photonic platform. By introducing a tensor-compressed BP-free training method, we have implemented a large-scale optical inference accelerator with significant hardware and energy reductions and an on-chip training framework that only requires additional inferences for gradient and derivative estimation, leading to scalable and robust optical PINN training. Through numerical simulations on a 20-dimensional Hamiltonian-Jacobi-Bellman (HJB) PDE, our method has shown impressive model size reduction (1.17×10^3 fewer MZIs), ultra-low-energy (1.36J) and ultra-high-speed (1.15s) PINN training. Future research includes further scaling up our PINN training framework, investigating high-speed MZI tuning methods, and demonstrating an electro-photonic integrated system for fJ/MAC high-speed PDE solvers.

References

[1] Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.

[2] MWMG Dissanayake and Nhan Phan-Thien. Neural-network-based approximations for solving partial differential equations. *communications in Numerical Methods in Engineering*, 10(3):195–201, 1994.

[3] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[4] Somil Bansal and Claire J Tomlin. Deepreach: A deep learning approach to high-dimensional reachability. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1817–1824, 2021.

[5] Derek Onken, Levon Nurbekyan, Xingjian Li, Samy Wu Fung, Stanley Osher, and Lars Ruthotto. A neural network approach applied to multi-agent optimal control. In *2021 European Control Conference (ECC)*, pages 1036–1041. IEEE, 2021.

[6] Xinling Yu, José EC Serrallés, Ilias I Giannakopoulos, Ziyue Liu, Luca Daniel, Riccardo Lattanzi, and Zheng Zhang. PIFON-EPT: Mr-based electrical property tomography using physics-informed fourier networks. *arXiv preprint arXiv:2302.11883*, 2023.

[7] Yichen Shen, Nicholas C Harris, Scott Skirlo, Mihika Prabhu, Tom Baehr-Jones, Michael Hochberg, Xin Sun, Shijie Zhao, Hugo Larochelle, Dirk Englund, et al. Deep learning with coherent nanophotonic circuits. *Nature photonics*, 11(7):441–446, 2017.

[8] J Feldmann, N Youngblood, M Karpov, H Gehring, X Li, M Stappers, M Le Gallo, X Fu, A Lukashchuk, A S Raja, J Liu, C D Wright, A Sebastian, T J Kippenberg, W H P Pernice, and H Bhaskaran. Parallel convolutional processing using an integrated photonic tensor core. *Nature*, 589(7840):52–58, 2021.

[9] 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*, 15(2):102–114, 2021.

[10] Michael Reck, Anton Zeilinger, Herbert J Bernstein, and Philip Bertani. Experimental realization of any discrete unitary operator. *Physical review letters*, 73(1):58, 1994.

[11] William R Clements, Peter C Humphreys, Benjamin J Metcalf, W Steven Kolthammer, and Ian A Walmsley. Optimal design for universal multiport interferometers. *Optica*, 3(12):1460–1465, 2016.

[12] 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 *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2020.

[13] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7583–7591, 2021.

[14] Matthew J Filipovich, Zhimu Guo, Mohammed Al-Qadasi, Bicky A Marquez, Hugh D Morison, Volker J Sorger, Paul R Prucnal, Sudip Shekhar, and Bhavin J Shastri. Silicon photonic architecture for training deep neural networks with direct feedback alignment. *Optica*, 9(12):1323–1332, 2022.

[15] Sonia Buckley and Adam McCaughan. A general approach to fast online training of modern datasets on real neuromorphic systems without backpropagation. In *Proceedings of the International Conference on Neuromorphic Systems 2022*, pages 1–8, 2022.

[16] Ilker Oguz, Junjie Ke, Qifei Wang, Feng Yang, Mustafa Yildirim, Niyazi Ulas Dinc, Jih-Liang Hsieh, Christophe Moser, and Demetri Psaltis. Forward-forward training of an optical neural network. *arXiv preprint arXiv:2305.19170*, 2023.

[17] Tyler W Hughes, Momchil Minkov, Yu Shi, and Shanhui Fan. Training of photonic neural networks through in situ backpropagation and gradient measurement. *Optica*, 5(7):864–871, 2018.

[18] Sunil Pai, Zhanghao Sun, Tyler W Hughes, Taewon Park, Ben Bartlett, Ian AD Williamson, Momchil Minkov, Maziyar Milanizadeh, Nathnael Abebe, Francesco Morichetti, et al. Experimentally realized in situ backpropagation for deep learning in photonic neural networks. *Science*, 380(6643):398–404, 2023.

[19] Xian Xiao, Mehmet Berkay On, Thomas Van Vaerenbergh, Di Liang, Raymond G Beausoleil, and SJ Ben Yoo. Large-scale and energy-efficient tensorized optical neural networks on iii–v-on-silicon moscap platform. *APL Photonics*, 6(12):126107, 2021.

[20] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[21] Ziyue Liu, Xinling Yu, and Zheng Zhang. Tt-pinn: a tensor-compressed neural pde solver for edge computing. *arXiv preprint arXiv:2207.01751*, 2022.

[22] Xian Xiao, Stanley Cheung, Sean Hooten, Yiwei Peng, Bassem Tossoun, Thomas Van Vaerenbergh, Geza Kurczveil, and Raymond G Beausoleil. Wavelength-Parallel Photonic Tensor Core Based on Multi-FSR Microring Resonator Crossbar Array. In *Optical Fiber Communication Conference*, page W3G.4, San Diego, CA, 2023.

[23] Yequan Zhao, Xinling Yu, Zhixiong Chen, Ziyue Liu, Sijia Liu, and Zheng Zhang. Tensor-compressed back-propagation-free training for (physics-informed) neural networks. *arXiv preprint arXiv:2308.09858*, 2023.

[24] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE transactions on automatic control*, 37(3):332–341, 1992.

[25] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[26] Sijia Liu, Pin-Yu Chen, Xiangyi Chen, and Mingyi Hong. signsgd via zeroth-order oracle. In *International Conference on Learning Representations*, 2019.

[27] 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. *Advances in neural information processing systems*, 32, 2019.

[28] Jiaqi Gu, Hanqing Zhu, Chenghao Feng, Zixuan Jiang, Ray Chen, and David Pan. L2ight: Enabling on-chip learning for optical neural networks via efficient in-situ subspace optimization. *Advances in Neural Information Processing Systems*, 34:8649–8661, 2021.

[29] Mehmet Berkay On, Yun-Jhu Lee, Xian Xiao, Roberto Proietti, and SJ Ben Yoo. Analysis of the hardware imprecisions for scalable and compact photonic tensorized neural networks. In *2021 European Conference on Optical Communication (ECOC)*, pages 1–4. IEEE, 2021.

[30] Ying Zhu, Grace Li Zhang, Bing Li, Xunzhao Yin, Cheng Zhuo, Huaxi Gu, Tsung-Yi Ho, and Ulf Schlichtmann. Countering variations and thermal effects for accurate optical neural networks. In *Proceedings of the 39th International Conference on Computer-Aided Design*, pages 1–7, 2020.

[31] Di Liang, Sudharsanan Srinivasan, Geza Kurczveil, Bassem Tossoun, Stanley Cheung, Yuan Yuan, Antoine Descos, Yingtao Hu, Zhihong Huang, Peng Sun, Thomas Van Vaerenbergh, Chong Zhang, Xiaoge Zeng, Songtao Liu, John E. Bowers, Marco Fiorentino, and Raymond G. Beausoleil. An energy-efficient and bandwidth-scalable dwdm heterogeneous silicon photonics integration platform. *IEEE Journal of Selected Topics in Quantum Electronics*, 28(6: High Density Integr. Multipurpose Photon. Circ.):1–19, 2022.