# PACE: Pacing Operator Learning to Accurate Optical Field Simulation for Complicated Photonic Devices

**Hanqing Zhu**[♥†], **Wenyan Cong**[♥], **Guojin Chen**[♥*], **Shupeng Ning**[♥], **Ray T. Chen**[♥], **Jiaqi Gu**[♦], **David Z. Pan**[♥‡]
[♦]Arizona State University
[♥]The University of Texas at Austin
[†]hqzhu@utexas.edu, [‡]dpan@ece.utexas.edu

## Abstract

Electromagnetic field simulation is central to designing, optimizing, and validating photonic devices and circuits. However, costly computation associated with numerical simulation poses a significant bottleneck, hindering scalability and turnaround time in the photonic circuit design process. Neural operators offer a promising alternative, but existing SOTA approaches,`NeurOLight`, struggle with predicting high-fidelity fields for real-world *complicated* photonic devices, with the best reported 0.38 normalized mean absolute error in `NeurOLight`. The interplays of highly complex light-matter interaction, e.g., scattering and resonance, sensitivity to local structure details, non-uniform learning complexity for full-domain simulation, and rich frequency information, contribute to the failure of existing neural PDE solvers. In this work, we boost the prediction fidelity to an unprecedented level for simulating complex photonic devices with a novel operator design driven by the above challenges. We propose a novel cross-axis factorized `PACE` operator with a strong long-distance modeling capacity to connect the full-domain complex field pattern with local device structures. Inspired by human learning, we further divide and conquer the simulation task for extremely hard cases into two progressively easy tasks, with a first-stage model learning an initial solution refined by a second model. On various *complicated* photonic device benchmarks, we demonstrate one sole `PACE` model is capable of achieving **73%** lower error with **50%** fewer parameters compared with various recent ML for PDE solvers. The two-stage setup further advances high-fidelity simulation for even more intricate cases. In terms of runtime, `PACE` demonstrates **154-577**× and **11.8-12**× simulation speedup over numerical solver using scipy or highly-optimized pardiso solver, respectively. **We open sourced the code and *complicated* optical device dataset at PACE-Light.**

## 1 Introduction

With advances in integrated photonics, photonic structures capable of transmitting or processing information are gathering increasing interest, fueled by the optical communication [24] and the recent resurgence of photonic analog computing [5, 20, 23, 36, 39, 40]. Light-empowered communication and computing offer a promising pathway for reshaping future AI systems, prompting the optical community to discover compact, customized devices [7, 32, 38] to overcome the limitations of bulky optical components. In this optical design process, numerical simulators, e.g., the popular finite difference frequency domain (FDFD) algorithm [11], is heavily used to obtain accurate optical fields for characterizing and optimizing device behavior. However, the significant time and computational

[*]This work was done when Guojin Chen was a visiting scholar at UT Austin.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

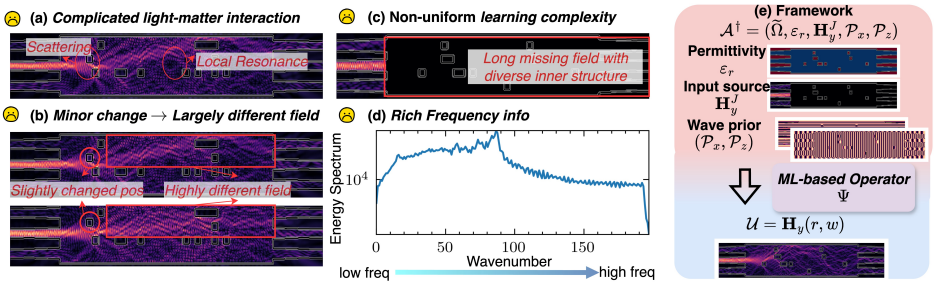



Figure 1: Challenges of complicated optical device simulation: (a-d) and learning framework (e).

costs associated with Maxwell partial differential equation (PDE) simulations, exacerbated by the need for finely tailored meshes and numerous simulation runs for iterative optimization, pose substantial bottlenecks in the design loop.

Recently, neural PDE solvers [3, 8, 9, 15, 16, 27, 33] have emerged as promising surrogate models for *fast* and *accurate* PDE solving. `NeurOLight` [8] represents the state-of-the-art (SOTA), extending neural operators to parametric photonic device simulations in a physics-agnostic manner. However, it still exhibits large errors in simulating real-world *complicated* optical devices, reporting a 0.38 normalized mean absolute error on the etched multi-mode interference (MMI) device [26]. One may wonder what the major challenges are, given the successes of neural operators in many scientific PDEs. Firstly, for *complicated* devices, the permittivity distribution is discrete and highly contrasting, transforming the Maxwell PDE into a multi-scale problem [1], further leading to complex light-matter interactions such as scattering and resonance, as illustrated in Fig. 1(a). Secondly, their optical fields are highly sensitive to local structural changes; even minor alterations can significantly impact the field, as depicted in Fig. 1(b). Moreover, with diversifying field patterns along the light propagation path, it shows non-uniform learning complexity especially in regions distant from the input light source. Finally, a spectral analysis provides insights into the frequency-domain challenges, as illustrated in Fig. 1(d). Unlike simpler systems where low frequencies dominate (e.g., Darcy flow shown in Fig. 10), *complicated* devices exhibit rich frequency spectra with high-frequency components. This diversity underpins the difficulty faced by previous neural PDE solvers in accurately simulating *complicated* photonic devices, supporting the assertion in [15] that no single model can universally solve all types of PDEs.

In this work, we tackle the challenging *real-world complicated* optical device simulation problem. We vastly boost prediction fidelity and keep **154-577**$\times$ and **11.8-12**$\times$ speedup over traditional numerical solver [11] on a 20-core CPU with `scipy` or highly-optimized `pardiso` solver, respectively.

Overall, we make the following key contributions:

- We introduce a novel cross-axis factorized `PACE` operator backbone, effectively capturing complex physical phenomena across the full domain in a parameter-efficient manner.
- We employ a divide-and-conquer approach inspired by human learning for extremely challenging cases, with a first-stage `PACE-I` to learn a rough approximation of the optical field, refined by a second-stage `PACE-II`.
- On various *complicated* device benchmarks, one sole `PACE` significantly outperforms baselines, achieving **73%** lower error with **50%** fewer parameters. Even compared to the best baseline, it lowers prediction error by over **39%** with **17%** fewer parameters. Our two-stage method further advances high-fidelity simulation for extremely challenging cases.
- We open-source the *complicated* optical device datatset and code at PACE-Light to facilitate AI for PDE community.

## 2 Preliminaries

### 2.1 Neural Operators for PDE

Recently, neural operators have emerged as a novel approach for developing machine learning models aimed at solving partial differential equations (PDEs). These models focus on learning

the mapping between the function spaces in a purely data-driven fashion. This holds the generalization capability within a family of PDEs and can potentially be adapted to different discretizations. Various function bases are utilized to build the operator learning model, such as the Fourier bases [16, 29, 2, 8], wavelet bases [9], spectral method [33], and attention layer [15, 3, 14]. These models have demonstrated remarkable performance and efficiency in solving specific types of problems, often achieving record-breaking results in certain applications. Despite their successes, it's important to recognize that the field of PDEs encompasses a wide variety of equations, each with its own unique properties and characteristics. As pointed out in recent research [15], there is no guarantee that a single type of data-driven model can effectively address all types of PDEs.

### 2.2 Optical Field Simulation with Machine Learning

Analyzing the propagation of light through optical devices is crucial for the optimization and design of photonic circuits. For a linear isotropic optical device, with a time-harmonic continuous-wave light beam shining on its input port, we can obtain the steady-state electromagnetic field distributions $\mathbf{E}(\boldsymbol{r}) = \hat{\mathbf{x}}\mathbf{E}_x + \hat{\mathbf{y}}\mathbf{E}_y + \hat{\mathbf{z}}\mathbf{E}_z$ and $\mathbf{H}(\boldsymbol{r}) = \hat{\mathbf{x}}\mathbf{H}_x + \hat{\mathbf{y}}\mathbf{H}_y + \hat{\mathbf{z}}\mathbf{H}_z$ by solving the steady-state frequency-domain *curl-of-curl* Maxwell PDE under absorptive boundary conditions [11],

$$\big((\mu_0^{-1}\nabla\times\nabla\times) - \omega^2\epsilon_0\epsilon_r(\boldsymbol{r})\big)\mathbf{E}(\boldsymbol{r}) = j\omega\mathbf{J}_e(\boldsymbol{r}),\ \big(\nabla\times(\epsilon_r^{-1}(\boldsymbol{r})\nabla\times) - \omega^2\mu_0\epsilon_0\big)\mathbf{H}(\boldsymbol{r}) = j\omega\mathbf{J}_m(\boldsymbol{r}) \quad (1)$$

where $\nabla\times$ is the curl operator, $\mu_0$ is the vacuum magnetic permeability, $\epsilon_0$ is the vacuum electric permittivity, $\epsilon_r$ is the relative electric permittivity, and $\mathbf{J}_m$ and $\mathbf{J}_e$ are the magnetic and electric current sources, respectively. The finite difference frequency domain (FDFD) method, a widely adopted numerical technique detailed in [11], is used to discretize these continuous-domain equations into an $M \times N$ mesh grid. This transforms the Maxwell PDEs into a linear system $\mathbf{AX} = \boldsymbol{b}$. Solving this system with a large sparse matrix $\mathbf{A} \in \mathbb{C}^{MN\times MN}$ is computationally expensive and challenging to scale. Although improvements have been made, such as replacing the `scipy` solver with the more efficient `pardiso` solver, the process remains prohibitively costly for large-scale applications.

Building neural networks (NNs) to accelerate this time-consuming simulation process has been investigated in predicting some key design parameters [26] or the entire optical field [30, 17, 4, 8]. `NeurOLight` extends the neural operator to optical field simulation, enabling learning a physics-agnostic parametric Maxwell PDE solver and achieving SOTA accuracy, while its performance on real-world *complicated* photonic device is still not satisfactory.

## 3 Understand the Problem Setup and Challenge

In this study, we aim to build a physics-agnostic neural operator $\Psi_\theta$ for parametric photonic device simulation in a data-driven fashion to approximate the ground-truth Maxwell PDE solver $\Psi^*: \mathcal{A} \to \mathcal{U}$ described in Eq. (1). Here, $\mathcal{U}$ represents the solution space for the optical field in $\mathbb{C}^{\Omega\times d_u}$ and $\mathcal{A} = (\Omega, \epsilon_r, \omega, \mathbf{J})$ represents the observation space of the Maxwell PDE, both defined over the continuous 2-D physical solving domain $\Omega = (l_x, l_z)$. We follow `NeurOLight` [8] to discretize the simulation domain $\Omega$ as $\widetilde{\Omega} = (M, N, \Delta l_x, \Delta l_z)$ with adaptive mesh granularity, i.e., with grid steps $\Delta l_x = l_x/M$ and $\Delta l_z = l_z/N$. Moreover, $(\widetilde{\Omega}, \epsilon_r, \omega)$ in the raw observation $\mathcal{A}$ is encoded as informative wave priors, $\mathcal{P}_z = e^{j\frac{2\pi\sqrt{\epsilon_r}}{\lambda}\mathbf{1}z^T\Delta l_z}$ and $\mathcal{P}_x = e^{j\frac{2\pi\sqrt{\epsilon_r}}{\lambda}x\mathbf{1}^T\Delta l_x}$, where $x = (0, 1, \cdots, M-1)$ and $z = (0, 1, \cdots, N-1)$, reflecting the propagation behaviors of light through different media. The input light source $\mathbf{J}$ is further modeled as a masked light source field $\mathbf{H}_y^J$.

Therefore, as illustrated in Fig. 1(e), the overarching objective is formulated as learning operator $\Psi_\theta$ that maps $\mathcal{A}^\dagger = (\epsilon_r, \mathbf{H}_y^J, \mathcal{P}_x, \mathcal{P}_z)$ to the target field $\mathcal{U}$ by optimizing the empirical error,

$$\theta^* = \min_\theta \mathbb{E}_{a\sim\mathcal{A}^\dagger}\big[\mathcal{L}\big(\Psi_\theta(a), u\big)\big], \quad (2)$$

### 3.1 Challenges in Predicting the Light Field of *Complicated* Photonic Devices

`NeurOLight` [8] delivers a pioneering effort in extending neural operators to the simulation of photonic devices, achieving SOTA accuracy. However, it still yields significant errors, particularly for real-world complicated devices, with a reported 0.38 normalized mean absolute error for etched MMI device [26, 12]. This leads us to an interesting reflection: despite the successes of neural

operators in solving scientific PDEs, why do they still fall short in *complicated* photonic device simulation? Below, we provide a detailed analysis that highlights the underlying learning challenges.

❶ **Complicated light-matter interaction in the optical field of real-world photonic device**. Permittivity $\epsilon_r$, a critical parameter in photonic devices, greatly impacts how light propagates through media. Designing new devices often involves manipulating the $\epsilon_r$ distribution across the domain. However, due to manufacturing limitations, $\epsilon_r$ changes are discrete rather than smooth. Moreover, researchers explore patterning materials with highly contrast permittivity to design compact devices [26, 31]. This discrete and highly contrasting permittivity transforms the Maxwell PDE into a *multiscale PDE problem* [1], with complicated light-matter interactions such as scattering resonance happening, shown in Fig. 1 (a), which has been shown difficult to predict from both scientific computing and operator learning perspectives [21, 35].

❷ **Significant prediction field variations from minor structural changes**. Due to the complex light-matter interactions within the field, even a slight change in the photonic structure can result in drastically different optical fields under the same input conditions, as shown in Fig. 1(b). This calls for a powerful backbone model that is capable of building the relationship between local rival changes with the global optical field transition.

❸ **Non-uniform learning difficulty along the spatial domain**. As shown in Fig. 1(c), with light shining in from a specific position and direction, it propagates through the media, resulting in non-uniform learning difficulties along the spatial domain. Due to the vast diversity of potential internal structures along the light propagation path, the light patterns are becoming highly diverse. Consequently, the data collected for training also incorporates the same phenomenon where many similar patterns are seen during training near the input sources, whereas the model faces more diverse patterns at greater distances. This makes it hard for the model to learn how to predict further regions, especially when the domain is elongated. This issue is analogous to the roll-out error encountered in temporal PDE modeling at the large time steps.

❹ **Rich frequency information lies in the predicted field**. We show the energy spectrum of the optical field in the frequency domain in Fig. 1 (d). The field, characterized by complex interactions such as scattering and resonance, exhibits rich frequency information, unveiling the learning complexity from a frequency-domain analysis. This confirms the usage of high-frequency modes in `NeurOLight`, underscoring the need for a parameter-efficient, robust, and powerful backbone model to resolve the parameter efficiency and overfitting issue with large modes.

## 4 Proposed `PACE` Methods

In this paper, we follow the standard operator learning model architecture as

$$a^{\dagger}(\boldsymbol{r}) \to v_0(\boldsymbol{r}) \to v_1(\boldsymbol{r}) \to \cdots v_K(\boldsymbol{r}) \to u(\boldsymbol{r}), \quad \forall \boldsymbol{r} \in \Omega. \tag{3}$$

We start with the convolutional stem used in [8] to project the PDE observation $a^{\dagger}(\boldsymbol{r})$ into a higher-dimensional feature space of dimension $C$. This is followed by a sequence of $K$ cascaded neural operator blocks, which gradually reconstruct the complex optical field within the $C$ dimensional space. At last, a head with two point-wise convolutional layers projects the $v_K(\boldsymbol{r})$ to the optical field space $u(\boldsymbol{r})$. Fig. 2(a) shows the proposed `PACE` neural operator block structure, formulated as,

$$v_{k+1}(\boldsymbol{r}) := \texttt{FFN}\big((\mathcal{K}v_k^{'})(\boldsymbol{r}) + v_k\big) + v_k, \ \forall \boldsymbol{r} \in \Omega; \ v_k^{'}(\boldsymbol{r}) = \text{pre-norm}(v_k(\boldsymbol{r})), \tag{4}$$

where $\mathcal{K}$ is the our proposed `PACE` operator and FFN$(\cdot)$ is a feedword network used in [8]. To stabilize the model performance when scaling to deeper layers, we add pre-normalization [34] and follow [13] to add a double skip. In this work, we consistently use the `NeurOLight` operator in the first two blocks to align our model with the horizontal and vertical wave prior encoding method adopted from `NeurOLight`, which we found slightly improves our accuracy.

### 4.1 Parameter-efficient and Effective Cross-axis Factorized `PACE` Operator

The neural operator design is key to obtaining satisfactory accuracy on a given PDE task. With the well-discussed challenges in Sec. 3.1, we derive key insights that have guided the development of our `PACE` operator in Fig. 2(b): (1) Long-distance full-domain modeling capacity, especially effectively modeling how local features impact the whole domain; (2) Isotropic model architecture with no

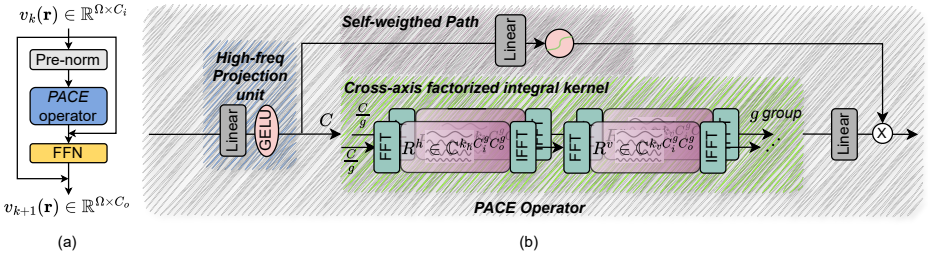



Figure 2: (a) PACE block with double skip and pre-normalization; (b) Our cross-axis factorized PACE operator.

down-sampling/ patching without losing local details; (3) Parameter efficiency under the needs of capturing high-frequency features.

Given the isotropic requirements, an operator based on Fourier bases is an ideal candidate as it achieves full-domain attention in the $O(nlogn)$ time complexity. However, the rich frequency information lying in the optical field requires the use of large frequency modes, making the FNO [16] with huge parameters and severe overfitting issues. NeurOLight [8] and Factorized FNO [29] propose to decompose the FNO block with independent 1-D FNO blocks in the full $N$-dimensional domain $\Omega$ (see Fig. 3), therefore, solving the parameter concern when utilizing high-frequency modes and serving as a regularization for overfitting. The only difference between NeurOLight and Factorized FNO [29] is whether they chunk the input or copy the input to the independent 1-D FNO block. We argue that their theoretical success is attributed to the *implicit full-domain integration* in Corollary 4.1.

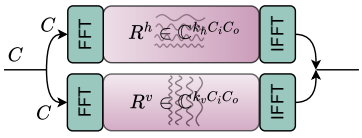



Figure 3: Factorized FNO [27, 8].

**Corollary 4.1.** *The factorized Fourier integral operator $\mathcal{K}$ [29, 8] factorizes the original Fourier integral operator [16] along each dimension $n$ in the N-dimension domain $\Omega$,*

$$(\mathcal{K}v_k)(\boldsymbol{r_1}) = \sum_n^N \mathcal{F}_n^{-1}(\mathcal{F}_n(\kappa_\phi^n) \cdot \mathcal{F}_n(\boldsymbol{r}_2))(\boldsymbol{r}_1), \quad \forall \boldsymbol{r}_1 \in \Omega, \tag{5}$$

*where each item explicitly computes a 1-D kernel integral, $\int_{\Omega_n} \kappa(\boldsymbol{r}_1, \boldsymbol{r}_2)^n v_k(\boldsymbol{r}_2)^n dv_k(\boldsymbol{r}_2)^n$. It implicitly implements full-domain kernel integration in $\Omega$ by stacking $\mathcal{K}$, i.e., $\mathcal{K}_0 \circ \mathcal{K}_1 \circ \cdots$,*

However, the reliance on implementing full-domain integration with multi-layers makes them *weak* operator candidates to achieve our first requirement, i.e., a *strong* model that is capable of building *full-domain* modeling between local structures with the global fields.

**Proposed cross-axis 2-D factorized integral kernel**. Aware of the above shortcomings of previous factorized FNO variants, in our 2-D domain, we propose to factorize the full domain integral in a cross-axis way along the horizontal (h) and vertical (v) axis:

$$\begin{aligned}(\mathcal{K}v_k)(\boldsymbol{r_1}) &= \int_\Omega \kappa(\boldsymbol{r}_1, \boldsymbol{r}_2) v_k(\boldsymbol{r}_2) \mathrm{d}v_k(\boldsymbol{r}_2), \quad \forall \boldsymbol{r}_1 \in \Omega, \\ &\approx \int_{\Omega_h} \kappa(\boldsymbol{r}_1, \boldsymbol{r}_2)^h \int_{\Omega_h} \kappa(\boldsymbol{r}_1, \boldsymbol{r}_2)^v v_k(\boldsymbol{r}_2) \mathrm{d}v_k(\boldsymbol{r}_2)^v \mathrm{d}v_k(\boldsymbol{r}_2)^h, \quad \forall \boldsymbol{r}_1 \in \Omega.\end{aligned} \tag{6}$$

This factorization enables an *explicit factorized full-domain integration*. It provides a *strong* way to capture the relationship between points in the domain $\Omega$, building the relationship between local structure with the complicated field pattern. The implementation of the above cross-axis integral can be efficiently implemented by Fourier Transform $\mathcal{F}(\cdot)$ when the kernel $\kappa(r_1, r_2) = \kappa(r_1 - r_2)$, as follows,

$$(\mathcal{K}v_k)(\boldsymbol{r_1}) = \mathcal{F}_h^{-1}(\mathcal{F}_h(\kappa^h) \cdot \mathcal{F}_h(\mathcal{F}_z^{-1}(\mathcal{F}_v(\kappa^v) \cdot \mathcal{F}_z(\boldsymbol{r}_2)))(\boldsymbol{r}_1), \quad \forall \boldsymbol{r}_1 \in \Omega, \tag{7}$$

in a $nlogn$ complexity ($n = MN$ in our 2-D cases with $\Omega \in \mathbb{C}^{M \times N}$).

**Group-wise cross-axis integration**. For input $\boldsymbol{r}$ with a channel dimension $C$, it can be viewed as the sampling of a set of functions $\{r_l(\cdot, \cdot)\}_{l=1}^C$ on grid point in the 2-D discretized domain $\Omega = \Omega_h \times$

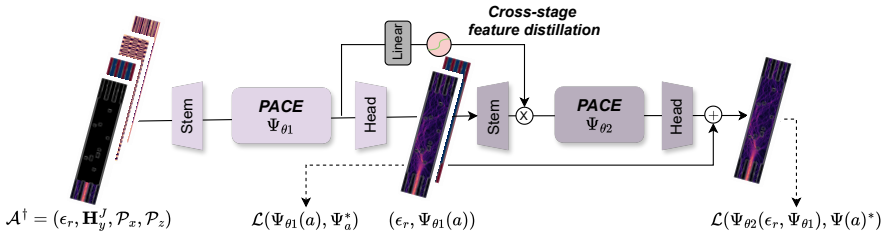



Figure 4: The proposed cascaded learning flow with two stages. The first stage learns an initial and rough solution, followed by the second stage to revise it further. A cross-stage distillation path is used to transfer the learned knowledge from the first stage to the second stage.

$\Omega_v$. The learnable integral kernel intrinsically performs information exchange along different grid points in $\Omega$. Similar to the multi-head design in Transformer, which assumes different heads extract different information, we can also partition the $C$ basis functions into $g$ disjoint sub-groups and feed each sub-group through our cross-axis factorized kernel. This grouping further reduces the number of parameters to $(\kappa_h + \kappa_v) \times \frac{C_o C_i}{g}$, showing significant parameter reduction compared to FNO ($\kappa_h \times \kappa_v \times C_o C_i$) and Factorized FNO ($(\kappa_h + \kappa_v) \times C_o C_i$), showing excellent parameter efficiency when utilizing large frequency modes is a must. We do an ablation study in Appendix A.4 to investigate the choices of different group $g$, where we find $g = 4$ strikes the best between parameter efficiency and model performance.

**Explicit projection unit $\xi$ for extracting high frequency information**. The optical field shows rich information in the frequency spectrum, reciting a special care of high-frequency information. Besides utilizing high-frequency modes, we propose to add an explicit projection module before the cross-axis integral, which is very simple as one linear layer followed by a non-linear activation, given non-linear activation is known to help generate high-frequency features [22].

**Self-weighted path for enhanced instance-based local feature attention**. The optical field's response is intricately linked to the minute variations in different photonic device structures. A self-weighted path is introduced to ensure the model can pay different attention to regions of significant influences for varying device structures. An instance-based weight is generated by passing the feature map after the projection unit through a linear layer and a `Sigmoid` unit, and then multiplied with the results after the cross-axis integral unit to provide instance-based attention.

Overall, the above ingredients are assembled together as our proposed `PACE` operator, as shown in Fig. 2 (b), which implements a self-weighted 2-D cross-axis factorized integral transform.

### 4.2 Cascaded Learning from Rough to Clear

With the effective `PACE` operator design, the prediction fidelity can be largely improved by only using a 12-layer `PACE` model (see Section. 5.2.1). But for some complicated benchmarks (e.g., etched MMI 3x3/5x5), it still yields $\sim$ 10% mean squared error, which is not satisfying. A straightforward solution might involve scaling up the model size, expecting additional layers would enhance performance. However, as demonstrated in [27], scaling to deep layers shows saturated performance after exceeding a specific number.

Existing ML for PDE solving work typically learns a model in a one-shot way by directly learning the underlying relationship from input-output pairs. Unlike AI systems, humans don't learn new and difficult tasks in a one-shot manner; instead, they learn skills progressively, starting with easier tasks and gradually moving to harder ones. For example, instead of directly learning how to solve equations, students first learn basic operations, such as addition and multiplication, and then move on to solving complex equations.

Hence, inspired by this human learning process, unlike previous work that directly learns a one-stage model, we propose to divide the challenging optical field prediction problem into two sequential latent tasks. The first task, undergoing the same problem setup as discussed in Sec. 3, could predict an initial, rough optical field based on the less informative raw PDE observation (we only have the light source and device permittivity distribution). Then, the successive second task could refine the rough prediction further by capturing more details and nuances, by accepting the predicted field $\Psi_{\theta_1}$ and

device permittivity $\epsilon_r$ as the input. Therefore, we assign *higher Fourier modes* to enable sufficient capacity. The divide-and-conquer way results in a cascaded two-stage model architecture, as shown in Fig. 4. The cascaded learning model is trained jointly (`PACE-I + PACE-II`) with the optimization target as the sum of two losses $\mathcal{L}(\Psi_{\theta 1}(a), u) + \mathcal{L}(\Psi_{\theta 2}(\Psi_{\theta 1}(a), \epsilon_r), u)$, where the first $\mathcal{L}(\Psi_{\theta 1}(a), u)$ serves as intermediate supervision that enfores the first stage model condensate the learned knowledge. To better connect the two-stage model, we propose a *cross-stage feature distillation path* to distill learned feature from the previous stage to the last by using a simple `Linear`→`Sigmoid` path.

# 5 Experimental Results

## 5.1 Experimental Setup

**Benchmarks:** We evaluate our methods on real-world *complicated* photonic devices that pose significant simulation challenges for ML surrogate models. This includes the Etched MMI with randomly placed rectangular cavities, used in [8], and the metaline device [37, 19] featuring two layers of randomly dimensioned meta-atoms. **These devices present a highly discrete and contrast permittivity distribution and complex light-matter interactions, making them ideal for testing the effectiveness of our model.** We generate our datasets using the open-source 2-D FDFD simulator, Angler [11], with generation details in Appendix A.1.

**Baselines**: We evaluate the proposed `PACE` model against a range of baselines, including the SOTA neural operator work, `NeurOLight` [8], for optical simulation. We also include representative operator learning models for scientific PDEs based on Fourier bases(FNO [16], Factorized FNO (F-FNO) [28, 29], U-NO [2], tensorized FNO (TFNO) [13]), attention kernels [15], and the latent spectral method (LSM) [33]. We also incorporate UNet [17, 4] and Dilated ResNet (Dil-ResNet) [25]. For a fair comparison, we keep a model size budget of under/near 4 million (M) parameters for baselines, except LSM [33] where the original implementation is adopted. Details on model configurations are in the Appendix A.3.

**Training setting and metric**: All models undergo training for 100 epochs using the AdamW optimizer with a weight decay of $1e^{-5}$ in a batch size of 4. To balance the optimization among different fields, we use normalized mean squared error (N-MSE) as the learning objective,

$$\mathcal{L}\big(\Psi_\theta(a), \Psi^*(a)\big) = (\|\Psi_\theta(\mathcal{E}(a)) - \Psi^*(a)\|^2)/\|\Psi^*(a)\|^2. \tag{8}$$

We don't use the previously-used mean absolute error (MAE) [8] as the metric given for complex-valued optical fields; we argue that L2 distance is a more accurate metric to evaluate the distance in the complex plane with a detailed analysis in Appendix A.6. We adopt the superposition-based mix-up technique [8] to generate input light combinations randomly to augment training data.

## 5.2 Main Results

### 5.2.1 Prediction Quality of Single `PACE` Model

In Tab. 1, we compare our 12-layer `PACE` model with various baselines on multiple real-world device benchmarks, showing significant **73.85%** smaller test error with **51.67%** fewer parameters on average. Notably, even when compared to the *best* baseline, 16-layer `NeurOLight`, we show over **39%** lower test error with over **17%** fewer parameters. Given the challenge ❷ that *trial structure change can totally change the optical field*, model relying on downsampling or patching fails to capture the local details, confirming the failure of the UNet and Transformer model. Moreover, the challenge ❶ and challenge ❸ call for a powerful model with long-distance modeling capability. Although Dil-ResNet utilizes a dilated block to enlarge the receptive field, it is insufficient for a large domain, validated by the result that it shows much better accuracy on the small Metaline than the etched MMI3x3. Capturing long-range dependency with the Fourier operator provides an efficient way to the isotropic model without any downsampling, therefore making the Fourier-operator type model show consistently better accuracy than other baseline methods. However, due to the challenge ❹ that there is rich frequency information in the predicted field, FNO-2d falls short due to the impediment of utilizing large modes given the large parameter count. We also compared it with the tensorized FNO 2d. However, we find the general tensor decomposition hurt the accuracy of this challenging task. `NeurOLight` shares a similar insight of Factorized FNO by factoring Fourier kernel with several independent 1-D Fourier kernels; however, as we argued before, it fails to establish

Table 1: Comparison of # parameters, training error (last epoch), and test error on three benchmarks among our PACE and various baselines. We use geo-means to report overall improvements across different benchmarks.

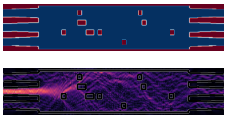
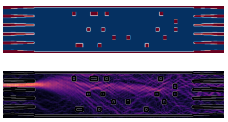
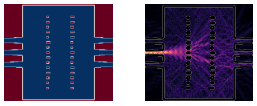
| Benchmarks | Model | #Params (M) ↓ | Train Err ($10^{-2}$) ↓ | Test Err ($10^{-2}$) ↓ |
|---|---|---|---|---|
| Etched MMI 3x3  | UNet [17, 4] | 3.88 | 63.03 | 65.32 |
| | Dil-ResNet [25] | 4.17 | 51.34 | 51.79 |
| | Attention-based model [15] | 3.75 | 70.05 | 69.85 |
| | U-NO [2] | 4.38 | 34.22 | 42.86 |
| | Latent-spectral method [33] | 4.81 | 55.07 | 55.16 |
| | FNO-2d [16] | 3.99 | 32.51 | 38.71 |
| | Tensorized FNO-2d [13] | 2.25 | 35.52 | 36.61 |
| | Factorized FNO-2d [29] | 4.02 | 24.2 | 32.81 |
| | NeurOLight [8] | 2.11 | 15.58 | 17.21 |
| | PACE | **1.71** | **9.51** | **10.59** |
| Etched MMI 5x5  | UNet [17, 4] | 3.88 | 65.73 | 66.01 |
| | Attention-based model [15] | 3.75 | 74.16 | 74.20 |
| | U-NO [2] | 4.38 | 37.92 | 42.24 |
| | Latent-spectral method [33] | 4.81 | 53.9 | 54.01 |
| | FNO-2d [16] | 3.99 | 33.12 | 36.49 |
| | Tensorized FNO-2d [13] | 2.25 | 39.11 | 39.45 |
| | Factorized FNO-2d [29] | 4.02 | 22.18 | 26.06 |
| | NeurOLight [28] | 2.11 | 18.04 | 17.41 |
| | PACE | **1.71** | **11.66** | **11.91** |
| Metaline 3x3  | UNet [17, 4] | 3.88 | 39.12 | 39.61 |
| | Dil-ResNet [25] | 4.17 | 12.37 | 13.20 |
| | Attention-based model [15] | 3.75 | 63.99 | 64.10 |
| | U-NO [2] | 4.38 | 19.27 | 22.09 |
| | Latent-spectral method [33] | 4.81 | 31.60 | 31.94 |
| | FNO-2d [16] | 3.21 | 19.73 | 20.88 |
| | Tensorized FNO-2d [13] | 1.58 | 30.60 | 31.04 |
| | Factorized FNO-2d [29] | 2.68 | 8.51 | 9.28 |
| | NeurOLight [8] | 1.49 | 6.76 | 6.09 |
| | PACE | **1.24** | **3.32** | **2.82** |
| Improvement over best baseline NeurOLight [8] | | **-17.70%** | **-41.23%** | **-39.03%** |
| Improvement over all baselines | | **-51.67%** | **-72.57%** | **-73.85%** |

Table 2: Comparison between our two-stage model and simply scaling more layers. All models use the same Fourier modes setup.

| Benchmarks | Model | Cross-stage dist. | #Params (M) ↓ | Train Err ($10^{-2}$) ↓ | Test Err ($10^{-2}$) ↓ |
|---|---|---|---|---|---|
| Etched MMI 3x3 | PACE-12 layer | - | 1.73 | 9.51 | 10.59 |
| | PACE-20 layer | - | 3.135 | 6.46 | 7.04 |
| | PACE-I + PACE-II | ✘ | 3.151 | 4.66 | 5.83 |
| | PACE-I + PACE-II | ✔ | 3.151 | 4.14 | 5.32 |
| Etched MMI 5x5 | PACE-12 layer | - | 1.73 | 11.66 | 11.91 |
| | PACE-20layer | - | 3.135 | 7.74 | 7.88 |
| | PACE-I + PACE-II | ✘ | 3.151 | 6.17 | 6.78 |
| | PACE-I + PACE-II | ✔ | 3.151 | 5.43 | 6.15 |

a strong full-domain modeling capacity by linking local details to the global complex field. Overall, our PACE block benefits from a physically meaningful cross-axis Fourier kernel factorization, equipping the capacity to capture full-domain dependency in a parameter-efficient way. Visualization of predicted results is in Appendix A.10.

### 5.2.2 Quality Improvement with Two-stage Model

We further compare the proposed cascaded two-stage model with the common practice of solely increasing # layers. We set the PACE-I as a 12-layer PACE model with Fourier modes(#Mode =70, #Mode =40), and PACE-II as a 8-layer PACE model with larger Fourier modes (#Mode =100, #Mode =40). As shown in Tab. 2, the two-stage setup introduces slight overhead for one extra set of stem and head but shows a clear margin over only increasing the number of layers in terms of both train error and test error. The cross-stage feature distillation further provides meaningful guidance by transferring learned features to the second-stage model, leading to the best accuracy for the two-stage setup. In Appendix A.7, we also show that the cross-stage distillation trick can improve model accuracy, similar to a more costly training setup, by training the two-stage models sequentially.

### 5.2.3 Speedup over Numerical Tools

To develop a fast surrogate ML model that can replace the Maxwell PDE solver, it's crucial to evaluate the speed-up of our PACE model compared to the FDFD numerical simulator Angler [11]. We vary the simulation domain size and set the grid step to 0.05 nm, scaling the discretized size pardiso linear solvers, respectively and number of frequency modes to ensure the model has sufficient capacity to capture the entire simulation domain. For comparison, we employ a 20-layer joint PACE model. As shown in Fig. 5, our PACE model achieves a speed-up of 150-577× and 12× over Angler on a 20-core Intel i7-12700 CPU using the scipy and We further set a larger simulation granularity, 0.075 nm, to check speedup if we tolerate simulation quality loss in commercial tools. However, we find that setting a larger granularity results in a significantly different field, as qualitatively shown in reb-Fig.3, with a corresponding N-MSE error of 1.2. Even though in this case, PACE still shows a 5.1-10.6× speedup over pardiso-based Angler with much better fidelity.

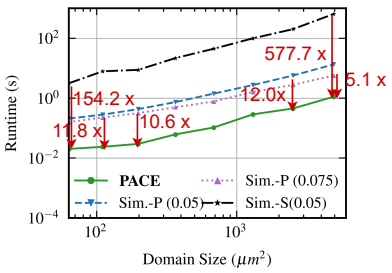



Figure 5: Speedup of PACE over angler [11] using scipy (S)/ pardiso (P) with simulation granularity (0.05nm) and (0.075nm).

### 5.3 Discussion

**Cross-axis PACE block design choices**. In Tab. 3, we *independently* alter individual components within the PACE operator to assess their effectiveness. The self-weighted path, which provides instance-specific weights, significantly improves model accuracy across various photonic device patterns. Removing this component results in a 17% increase in error, highlighting its importance. Similarly, eliminating the high-frequency projection unit leads to a 23% worse error, emphasizing its crucial role in capturing high-frequency features. To further illustrate this, we visualize the feature maps in the frequency domain before and after applying the nonlinear activation in the high-frequency projection unit. As shown in Fig. 11, the nonlinear activation effectively amplifies high-frequency components, supporting our claim and validating the design decision to incorporate an additional high-frequency projection path. Lastly, we replace our cross-axis Factorized integral kernel with a recent tensorized FNO (TFNO) [13] (tucker decomposition with rank 0.02). While TFNO effectively models long-range dependencies, matching our parameter count required aggressive decomposition, which significantly degraded performance. This comparison underscores the advantage of our physically grounded *cross-axis factorized kernel*.

Table 3: Model design ablation on Metaline dataset.

| Variants | #Params (M)↓ | #Train Err ($10^{-2}$)↓ | #Test Err ($10^{-2}$)↓ |
|---|---|---|---|
| **8-layer PACE** | **0.82** | **5.65** | **4.82** |
| No self-weighted path | 0.8 | 6.33 | 5.66 (+0.84) |
| No projection unit | 0.8 | 6.58 | 5.97 (+1.15) |
| Use TFNO | 1.06 | 10.80 | 9.51 (+4.69) |

**Generalization to out-of-distribution testing**. As an operator model that is parameter-agnostic, it is important to test the generalization for out-of-distribution data with unseen parameters. We re-generate photonic devices with different device configurations (size, etched region, etc.) and unseen frequencies in our interested wavelength range (1.53-1.565 $\mu$m), i.e., C-band. As shown in Fig. 6, our PACE model generalizes well on unseen simulation frequency and new devices. It is a vital test to prove the usefulness of PACE in helping device design within an interested wavelength range. We also test the accuracy outside the C-band, where PACE shows good accuracy on neighboring wavelengths while holding a 10-15% error at a further range. This is expected since wave propagation is sensitive to frequency. It can be mitigated by incorporating sampled wavelengths into training.

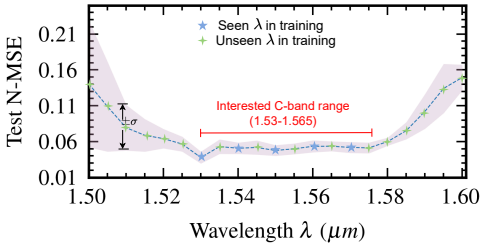



Figure 6: Generalize to unseen wavelength in interested C-band (1.53-1.565) and outside C-band.

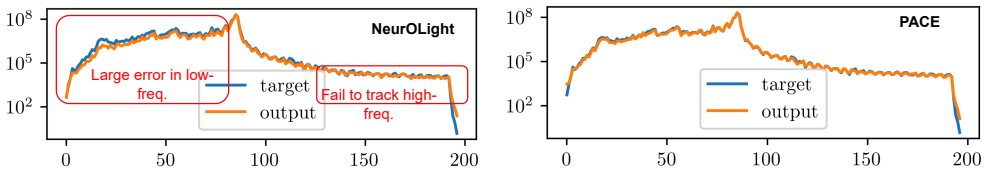



Figure 8: The radial energy spectrum of predicted fields from NeurOLight and PACE. NeurOLight fails to align precisely with the targeted field in both low-frequency and high-frequency parts.

**Are `PACE` a general enhancer module for Fourier-type operator?** We further investigate whether our new `PACE` operator is a general enhancer for other Fourier operators, rather than a dedicated module for our own model architecture. We randomly insert four `PACE` blocks into Facztoried FNO [27] and test the error on Metaline3x3 and Etched MMI 3x3 benchmarks, showing up to 28% error reduction as shown in Fig. 7 with much fewer parameters.

**Comparison with operator for multi-scale PDE**. Noticing that our problem shares similar complexities in solving multi-scale PDEs with neural operator [18, 35], we further compare our approach with the recent method [35] that alternates Fourier operator with dilated convolution layer to better capture local details. On the etched MMI 3x3 dataset, we implement a 14-layer model with alternating `NeurOLight` block and dilated convolution layer. It yields a 1.73 M parameter count similar to our `PACE` but shows a 17.4 N-MSE error, much worse than ours (10.59).

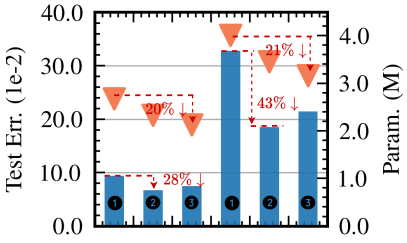



Figure 7: Insert 4 `PACE` module (❷: g=2; ❸: g=4) randomly in Factorized FNO(❶).

**Spectrum of the predicted field**: The predicted field spectrums of PACE and `NeurOLight` are in Fig. 8. Although `NeurOLight` uses the same frequency modes, it fails to align well with both the low-frequency and high-frequency regions. `PACE` excellently aligns with the baseline spectrum compared to NeurOLight,

## 6 Conclusion

In this work, we *pace* the simulation fidelity on highly challenging *complicated* photonic devices to an unprecedented level. Our novel cross-axis factorized `PACE` operator enables the neural PDE solver to capture complex relationships between local device structures and the resulting complex optical field across the entire simulation domain. Furthermore, we introduce a cascaded two-stage learning paradigm to further enhance the prediction quality when one sole `PACE` is not sufficient, demonstrating better quality enhancement than simply adding more layers. Experiments demonstrate that `PACE` achieves a remarkable 73% reduction in error with 50% fewer parameters compared to previous methods. Our method also offers significant speedup (11.8x to 577x) over traditional numerical solvers. Looking forward, we aim to integrate our model into the design optimization loop for photonic devices and circuits. Moreover, we want to emphasize that our proposed operator and learning strategy are not dedicated to photonic cases but generally applied to challenging PDE problems with similar problem characteristics, e.g., multi-scale PDE problems.

**Limitations and Broader Impact.** This work focuses on steady-state optical field solutions using the FDFD method. Exploring the effectiveness of operator learning for the Finite-Difference Time Domain (FDTD) can be an interesting direction. Moreover, the FFT kernels on GPU are not fully optimized [6]. Employing specialized, optimized FFT kernels can unlock even greater computational efficiency on GPUs, further accelerating the neural PDE solver.

## 7 Acknowledgments and Disclosure of Funding

We acknowledge NVIDIA for donating its A100 GPU workstations and the support from TILOS, NSF-funded National Artificial Intelligence Research Institute. Additionally, this work was supported by the Air Force Office of Scientific Research (AFOSR) through the AFOSR project, contract FA9550-23-1-0452, and the Multidisciplinary University Research Initiative (MURI) program under contract No. FA9550-17-1-0071.

## References


[1] Habib Ammari, Yves Capdeboscq, and Hyeonbae Kang. *Multi-scale and High-contrast PDE: from Modelling, to Mathematical Analysis, to Inversion*, volume 577. American Mathematical Society, 2012.

[2] Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *arXiv e-prints*, pages arXiv–2204, 2022.

[3] Shuhao Cao. Choose a transformer: Fourier or galerkin. *Advances in neural information processing systems*, 34:24924–24940, 2021.

[4] Mingkun Chen, Robert Lupoiu, Chenkai Mao, Der-Han Huang, Jiaqi Jiang, Philippe Lalanne, and Jonathan Fan. Physics-augmented deep learning for high-speed electromagnetic simulation and optimization. *Nature*, 2021.

[5] Johannes Feldmann, Nathan Youngblood, Maxim Karpov, Helge Gehring, Xuan Li, Maik Stappers, Manuel Le Gallo, Xin Fu, Anton Lukashchuk, Arslan Raja, Junqiu Liu, David Wright, Abu Sebastian, Tobias Kippenberg, Wolfram Pernice, and Harish Bhaskaran. Parallel convolutional processing using an integrated photonic tensor core. *Nature*, 2021.

[6] Daniel Y. Fu, Hermann Kumbong, Eric Nguyen, and Christopher Ré. FlashFFTConv: Efficient convolutions for long sequences with tensor cores. 2023.

[7] Jiaqi Gu, Chenghao Feng, Hanqing Zhu, Zheng Zhao, Zhoufeng Ying, Mingjie Liu, Ray T Chen, and David Z Pan. Squeezelight: A multi-operand ring-based optical neural network with cross-layer scalability. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(3):807–819, 2022.

[8] Jiaqi Gu, Zhengqi Gao, Chenghao Feng, Hanqing Zhu, Ray Chen, Duane Boning, and David Pan. Neurolight: A physics-agnostic neural operator enabling parametric photonic device simulation. *Advances in Neural Information Processing Systems*, 35:14623–14636, 2022.

[9] Gaurav Gupta, Xiongye Xiao, and Paul Bogdan. Multiwavelet-based operator learning for differential equations. In *Proc. NeurIPS*, 2021.

[10] Jayesh K Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling. *arXiv preprint arXiv:2209.15616*, 2022.

[11] Tyler W. Hughes, Momchil Minkov, Ian A. D. Williamson, and Shanhui Fan. Adjoint method and inverse design for nonlinear nanophotonic devices. *ACS Photonics*, 2018.

[12] Junhyeong Kim, Berkay Neseli, Jae yong Kim, Jinhyeong Yoon, Hyeonho Yoon, Hyo hoon Park, and Hamza Kurt. Inverse design of an on-chip optical response predictor enabled by a deep neural network. *Opt. Express*, 2023.

[13] Jean Kossaifi, Nikola Kovachki, Kamyar Azizzadenesheli, and Anima Anandkumar. Multi-grid tensorized fourier neural operator for high-resolution pdes. *arXiv preprint arXiv:2310.00120*, 2023.

[14] Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023.

[15] Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for pde surrogate modeling. *Advances in Neural Information Processing Systems*, 36, 2024.

[16] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[17] Joowon Lim and Demetri Psaltis. Maxwellnet: Physics-driven deep neural network training based on maxwell's equations. *Appl. Phys. Lett.*, 2022.

[18] Xinliang Liu, Bo Xu, and Lei Zhang. Ht-net: Hierarchical transformer based operator learning model for multiscale pdes. 2022.

[19] Nina Meinzer, William L Barnes, and Ian R Hooper. Plasmonic meta-atoms and metasurfaces. *Nature photonics*, 8(12):889–898, 2014.

[20] Shupeng Ning, Hanqing Zhu, Chenghao Feng, Jiaqi Gu, Zhixing Jiang, Zhoufeng Ying, Jason Midkiff, Sourabh Jain, May H Hlaing, David Z Pan, et al. Photonic-electronic integrated circuits for high-performance computing and ai accelerators. *Journal of Lightwave Technology*, 2024.

[21] Mario Ohlberger and Barbara Verfurth. A new heterogeneous multiscale method for the helmholtz equation with high contrast. *Multiscale Modeling & Simulation*, 16(1):385–411, 2018.

[22] Bogdan Raonic, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes. *Advances in Neural Information Processing Systems*, 36, 2024.

[23] 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.

[24] Yaocheng Shi, Yong Zhang, Yating Wan, Yu Yu, Yuguang Zhang, Xiao Hu, Xi Xiao, Hongnan Xu, Long Zhang, and Bingcheng Pan. Silicon photonics for high-capacity data communications. *Photonics Research*, 10(9):A106–A134, 2022.

[25] Kim Stachenfeld, Drummond Buschman Fielding, Dmitrii Kochkov, Miles Cranmer, Tobias Pfaff, Jonathan Godwin, Can Cui, Shirley Ho, Peter Battaglia, and Alvaro Sanchez-Gonzalez. Learned simulators for turbulence. In *International conference on learning representations*, 2021.

[26] Mohammad H. Tahersima, Keisuke Kojima, Toshiaki Koike-Akino, Devesh Jha, Bingnan-Wang, and Chungwei Lin. Deep neural network inverse design of integrated photonic power splitters. *Sci. Rep.*, 2019.

[27] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*, 2021.

[28] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *NeurIPS Workshop*, 2021.

[29] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023.

[30] Rahul Trivedi, Logan Su, Jesse Lu, Martin F. Schubert, and JelenaVuckovic. Data-driven acceleration of photonic simulations. *Sci. Rep.*, 2019.

[31] Barbara Verfürth. Heterogeneous multiscale method for the maxwell equations with high contrast. *ESAIM: Mathematical Modelling and Numerical Analysis*, 53(1):35–61, 2019.

[32] Zi Wang, Lorry Chang, Feifan Wang, Tiantian Li, and Tingyi Gu. Integrated photonic metasystem for image classifications at telecommunication wavelength. *Nature communications*, 13(1):2131, 2022.

[33] Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional pdes with latent spectral models. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[34] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR, 2020.

[35] Bo Xu and Lei Zhang. Dilated convolution neural operator for multiscale partial differential equations. 2023.

[36] Xingyuan Xu, Mengxi Tan, Bill Corcoran, Jiayang Wu, Andreas Boes, Thach G. Nguyen, Sai T. Chu, Brent E. Little, Damien G. Hicks, Roberto Morandotti, Arnan Mitchell, and David J. Moss. 11 TOPS photonic convolutional accelerator for optical neural networks. *Nature*, 2021.

[37] Sanaz Zarei, Mahmood-reza Marzban, and Amin Khavasi. Integrated photonic neural network based on silicon metalines. *Optics Express*, 28(24):36668–36684, 2020.

[38] H. H. Zhu, J. Zou, H. Zhang, Y. Z. Shi, S. B. Luo, et al. Space-efficient optical computing with an integrated chip diffractive neural network. *Nature Communications*, 2022.

[39] Hanqing Zhu, Jiaqi Gu, Hanrui Wang, Zixuan Jiang, Zhekai Zhang, Rongxing Tang, Chenghao Feng, Song Han, Ray T Chen, and David Z Pan. Lightening-transformer: A dynamically-operated optically-interconnected photonic transformer accelerator. In *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 686–703. IEEE, 2024.

[40] Hanqing Zhu, Keren Zhu, Jiaqi Gu, Harrison Jin, Ray T Chen, Jean Anne Incorvia, and David Z Pan. Fuse and mix: Macam-enabled analog activation for energy-efficient neural acceleration. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pages 1–9, 2022.

# A Appendix

## A.1 Dataset Generation

We generate our customized etched MMI and Metaline dataset using the open-source FDFD simulator angler [11]. For each type of device, we random sample 5.12 K device configuration following the Tab. 4, and generate single-source data by sweeping the input light over the input ports. We randomly sample the device's physical dimension, input/output waveguide width and input light source frequencies. For etched MMIs, we randomly sample etched cavity sizes, ratios (which determine the number of cavities in the MMIs), and permittivities in the controlling region. For Metaline, we randomly sample the metaatom physical dimension with a fixed total number of 20.

We discretize the domain of etched MMI by $80 \times 384$, and the domain of metaline by $128 \times 144$.

Table 4: Summary of etched MMI device design variable's sampling range, distribution, and unit.

| Variables | Value/Distribution | | | Unit |
|---|---|---|---|---|
| | Metaline $3 \times 3$ | Etched MMI $3 \times 3$ | Etched MMI $5 \times 5$ | |
| Length | $\mathcal{U}(8, 10)$ | $\mathcal{U}(20, 30)$ | $\mathcal{U}(25, 35)$ | $\mu m$ |
| Width | $\mathcal{U}(10, 12)$ | $\mathcal{U}(5.5, 7)$ | $\mathcal{U}(7.5, 9)$ | $\mu m$ |
| Port Length | 1.5 | 1.5 | 1.5 | $\mu m$ |
| Port Width | $\mathcal{U}(0.5, 0.8)$ | $\mathcal{U}(0.8, 1.1)$ | $\mathcal{U}(0.8, 1.1)$ | $\mu m$ |
| Taper Length | 3 | 4.5 | 4.5 | $\mu m$ |
| Taper Width | 1.3 | 1.3 | 1.3 | $\mu m$ |
| Border Width | 0.25 | 0.25 | 0.25 | $\mu m$ |
| PML Width | 1.5 | 1.5 | 1.5 | $\mu m$ |
| Wavelengths $\lambda$ | $\mathcal{U}(1.53, 1.565)$ | $\mathcal{U}(1.53, 1.565)$ | $\mathcal{U}(1.53, 1.565)$ | $\mu m$ |
| Cavity Ratio | - | $\mathcal{U}(0.05, 0.1)$ | $\mathcal{U}(0.05, 0.1)$ | - |
| Note | 2-layer random meta-atoms | random slots | random slots | - |
| Relative Permittivity $\epsilon_r$ | {2.07, 12.11} | {2.07, 12.11} | {2.07, 12.11} | - |

## A.2 Training Settings

We implement all models and training logic in PyTorch 2.3. We use A100 and A6000 to train our models and report the latency running on a single A100 GPU with `torch.compile`. For Benchmarking FDFD simulator performance, we use the Intel 12th Gen Intel(R) Core(TM) i7-12700 with 20 CPU cores. We split all single-source examples into 72% training data, 8% validation data, and 20% test data.

For training, we set the number of epochs to 100 with an initial learning rate of 0.002, cosine learning rate decay, and a mini-batch size of 4. We use adamW as the optimizer with the weight decay 1e-5 to avoid over-fitting. Moreover, we apply stochastic network depth with a linear scaling strategy.

## A.3 Model Designs

To ensure a comprehensive evaluation, we compare our proposed model against recently published and available SOTA baselines, encompassing various architectural paradigms such as the Fourier-operator models, attention-based models, and latent space methods. To maintain fairness in comparison, we constrain the parameter count of all models to be under 4 M in most cases and use open-sourced implementations.

Here, we report the model details for baselines for the etched MMI dataset.

**UNet**. We construct a 4-level convolutional UNet with a base channel number of 36, following the open-sourced implementation[2]. The total parameter count is 3.88 M.

**Dil-ResNet** [25]. We use the implementation in open-sourced pdearena [10][3], with a channel number of 128 and enabled normalization. The total parameter count for Dil-ResNet is 4.17 million.

[2]https://github.com/JeremieMelo/NeurOLight

[3]https://pdearena.github.io/pdearena/

**FNO-2d** [16]. We use 6 2-D FNO layers, and the Fourier modes are set to (#Mode =32, #Mode =10) for the etched MMI dataset and (#Mode =16, #Mode =16) for the Metaline dataset, resulting in the total parameter count as 3.99 M or 3.21 M. We use the implementation[4].

**Tensorized FNO-2d** [13]. One obvious advantage is that our model features low parameters. Hence, we will compare it with tensorized FNO, which compresses the model weights with the tensor decomposition method. We adopt the implementation in[5] and use the model designed for the darcy flow problem. We use 5 2-D FNO layers, and the Fourier modes are set to (#Mode =40, #Mode =20) for the etched MMI dataset and (#Mode =24, #Mode =24) for the Metaline dataset. Tucker decomposition is used with a rank of 0.42. The total parameter count is 2.25 M and 1.58 M, respectively, for the two types of datasets.

**F-FNO-2d**. For factorized Fourier neural operator (F-FNO), we use 13 F-FNO layers with a channel number of 52. The Fourier modes are set to (#Mode =70, #Mode =40) for etched MMI dataset and (#Mode =36, #Mode =36) for Metaline dataset, leading to the total parameter count as 4.02 M and 2.68M. The FNO-2d implementation is referred to[6]. We use the same projection head as ours.

**U-NO-2d** [2]. For U-shaped neural operators, we follow the implementation[7]. We use their 11-layer UNO with a base channel 24. The total parameter count is 4.38 M.

**Attention-based operator** [15]. For the attention-based neural operator, we choose the most recent Transformer-type model [15] and use its official implementation[8]. We use 3 layer-attention with 12 heads. The total dimension is 128. The total parameter count is 3.75 M.

**Latent Spectral Method** [33]. For the latent spectral method, we use the original implementation in[9]. The number of bases is set to 12, and the channel number is 32. The patch size is set to $4\times4$. The total parameter count is 4.8 M.

`NeuroLight` [8]. We use the same implementation[10] in the original paper with 16 layers. For the etched MMI dataset, the Fourier modes are set to (#Mode =70, #Mode =40). For the Metaline dataset, Fourier modes are set to (#Mode =36, #Mode =36). The total number of parameters is 2.11M and 1.49 M for the two cases.

`PACE`. For our proposed `PACE`, we use 12 layers, with the first two being the same factorized layers in [8], since we found it is important first to generate some meaningful wave patterns and then do global information swapping. The Fourier modes are set to (#Mode =70, #Mode =40). We use the same convolution stem in [8] to extract information before going through the feature propagator and the same projection head. The total number of parameters is 1.73M. For the second stage `PACE-II` model, we use 8 layers with all being `PACE` operators, where Fourier modes are set to (#Mode =100, #Mode =40) For the Metaline dataset, we solely use a 12-layer `PACE` with Fouier modes being (#Mode =36, #Mode =36).

### A.4 Ablation study on group number choices

We run an 8-layer `PACE` model on the Metaline dataset by setting the group size to 1, 2, 4, and show the train and test error in Tab. 5 We use #group=4 in our paper, which balances between parameter efficiency and test error.

### A.5 Ablation Study of Double Skip and Pre-Normalization

We further investigate whether the observed improvements in accuracy are attributed to the incorporation of double skip connections and pre-normalization, which were incorporated into our model to stabilize it in deeper layers with better generalization. We add these two techniques to `NeurOLight` and compare them with ours `PACE` in Tab.. 6. The double skip and pre-normalization can make the model generalize well for test data, while the training error is slightly improved as normalization

[4] https://github.com/JeremieMelo/NeurOLight
[5] https://github.com/neuraloperator/neuraloperator
[6] https://github.com/alasdairtran/fourierflow
[7] https://github.com/ashiq24/UNO/blob/main/navier_stokes_uno2d.py
[8] https://github.com/BaratiLab/FactFormer/tree/main
[9] https://github.com/thuml/Latent-Spectral-Models
[10] https://github.com/JeremieMelo/NeurOLight

Table 5: Ablation on # group on an 8-layer PACE model on the Metaline dataset.

| # Group | #Params (M)↓ | #Train Err ($10^{-2}$)↓ | #Test Err ($10^{-2}$)↓ |
|---|---|---|---|
| 1 | 2.15 | 4.89 | 4.55 |
| 2 | 1.27 | 5.33 | 4.80 |
| 4 | 0.825 | 5.65 | 4.82 |

Table 6: Ablation on the comparison between PACE and NeurOLight with the adopted double skip and pre-normalization.

| Model | Double skip & Pre-norm. | #Params ↓ | #Train Err ($10^{-2}$)↓ | #Test Err ($10^{-2}$)↓ |
|---|---|---|---|---|
| NeurOLight-16 layer | ✗ | 2108258 | 15.58 | 17.21 |
| NeurOLight-16 layer | ✔ | 2110306 | 15.26 | 15.87 |
| PACE-12 layer | ✗ | 1709026 | 10.32 | 11.06 |
| PACE-12 layer | ✔ | 1710562 | 9.60 | 10.60 |

can be seen as some linear affine. However, it still shows much worse accuracy than our model, especially given the context our model is shallower with fewer parameters.

### A.6 L2 distance is a more informative metric compared to L1 distance for distance evaluation

**Corollary A.1.** *Consider two complex numbers in polar form, $z_1 = r_1\angle\phi_1$ and $z_2 = r_2\angle\phi_2$. Their mean square error is rotation invariant, as shown by:*

$$|z_1, z_2|_2^2 = |r_1\cos\phi_1 - r_2\cos\phi_2|^2 + |r_1\sin\phi_1 - r_2\sin\phi_2|^2 \quad = r_1^2 + r_2^2 - 2r_1r_2\cos(\phi_1 - \phi_2). \tag{9}$$

*This distance metric depends solely on the difference in signal norm and angle between $z_1$ and $z_2$. However, their mean absolute error is the rotation variant:*

$$|z1, z_2|_1 = |r_1\cos\phi_1 - r_2\cos\phi_2| + |r_1\sin\phi_1 - r_2\sin\phi_2|. \tag{10}$$

In the complex plane, optimizing MAE equates to minimizing the summed L1 distances of the real and imaginary components. However, the L1 distance is rotation-variant. A simple rotation of the two complex numbers on the plane results in changing L1 distance, as shown in Fig. 9, while the true distance does not alter. Therefore, it is not an appropriate metric as it cannot accurately measure proximity in the complex plane. In this way, we use L2 distance in the loss (mean squared loss) that is rotation invariant, as proved in corollary A.1, which exactly captures the distance in the complex plane.

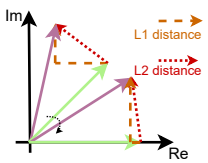



Figure 9: L1 and L2 distance in the complex plane.

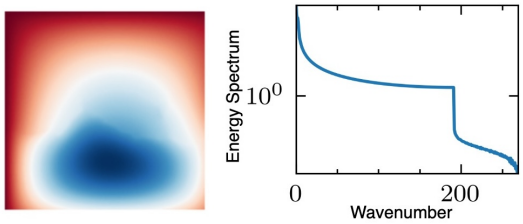



Figure 10: Radial energy spectrum for one solution of Darcy flow problem.

Table 7: Results of train two-stage model sequentially.

| Benchmarks | Model | Cross-stage dist. | #Params (M) ↓ | Train Err ($10^{-2}$) ↓ | Test Err ($10^{-2}$) ↓ |
|---|---|---|---|---|---|
| Etched MMI 3x3 | PACE-12 layer | - | 1.73 | 9.51 | 10.59 |
| | PACE-I → PACE-II | ✔ | 3.151 | 3.91 | 5.06 |
| Etched MMI 5x5 | PACE-12 layer | - | 1.73 | 11.66 | 11.91 |
| | PACE-I → PACE-II | ✔ | 3.151 | 5.51 | 6.15 |

### A.7 Train two-stage model sequentially

We also show the error by training our proposed two-stage model sequentially in Tab. 7, which shows a similar error to our joint training approach when equipping with our proposed cross-stage feature distillation.

Training sequentially is more costly than joint training, as second-stage training requires first inferencing with the first stage to get the predicted results.

### A.8 Visualization of energy spectrum

We generate the prediction field's radial energy spectrum by first transferring the image from the spatial domain to the spectral domain and then shifting the transferred image to the center. Then, the wavenumber is computed as the distance with respect to the center.

We sum the squared magnitude of the Fourier coefficients that fall into the specific number, which is implemented following open-sourced code [11].

We also visualize one example of darcy flow problem, as shown in Fig. 10. It shows highly distant characteristics compared to our optical field, with most information concentrating on low-frequency parts.

### A.9 Visualization of feature map before/after non-linear activation in our explicitly designed high-frequency projection path

We visualize the first 6 channels of feature maps before and after the nonlinear activation in the last PACE layer by showing them in the frequency domain. As shown in Fig. 11, the nonlinear activation can ignite high-frequency features, which confirms our claim and validates our design choice of injecting an extra high-frequency projection path.

### A.10 Visualization of prediction

We provide visualization figures on etched MMI 3x3 devices in Fig. 12 and metaline devices in Fig. 13. We provide the predicted fields $\Psi(a)$, the groud-truth field $\Psi(a)^*$ and the residual error $\Psi(a)^* - \Psi(a)$ of Dil-ResNet, Facztoried FNO, `NeurOLight` and our `PACE`. For etched MMI test cases, we show both the single 12-layer `PACE` model and the joint 20-layer model `PACE-I + PACE-II`. Our `PACE` shows much better prediction results with a near-black error map compared to other baseline methods.

[11] https://github.com/autonomousvision/projectedgan/blob/main/torch_utils/utils_spectrum.py

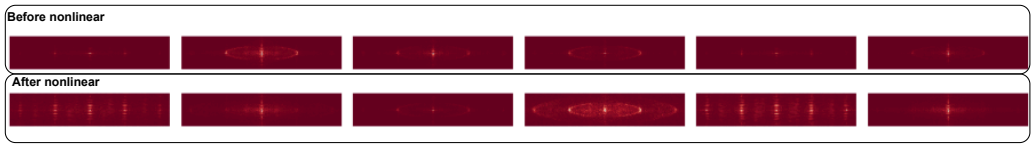



Figure 11: Frequency-domain visualization of feature map before and after non-linear activation in the last PACE block(The center represents low frequency). The pattern is shifted to the center to understand the frequency content better.

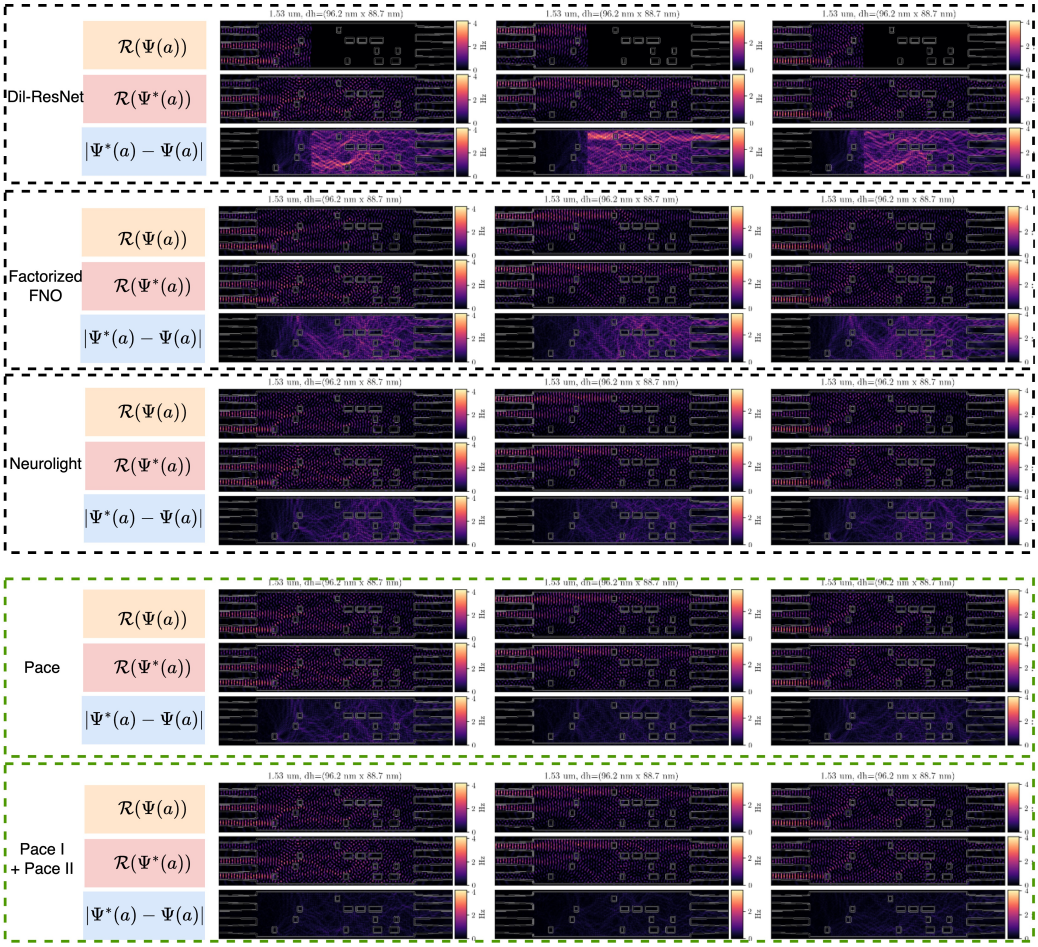



Figure 12: Visualization of test cases on etched MMI 3x3 devices with random input sources.

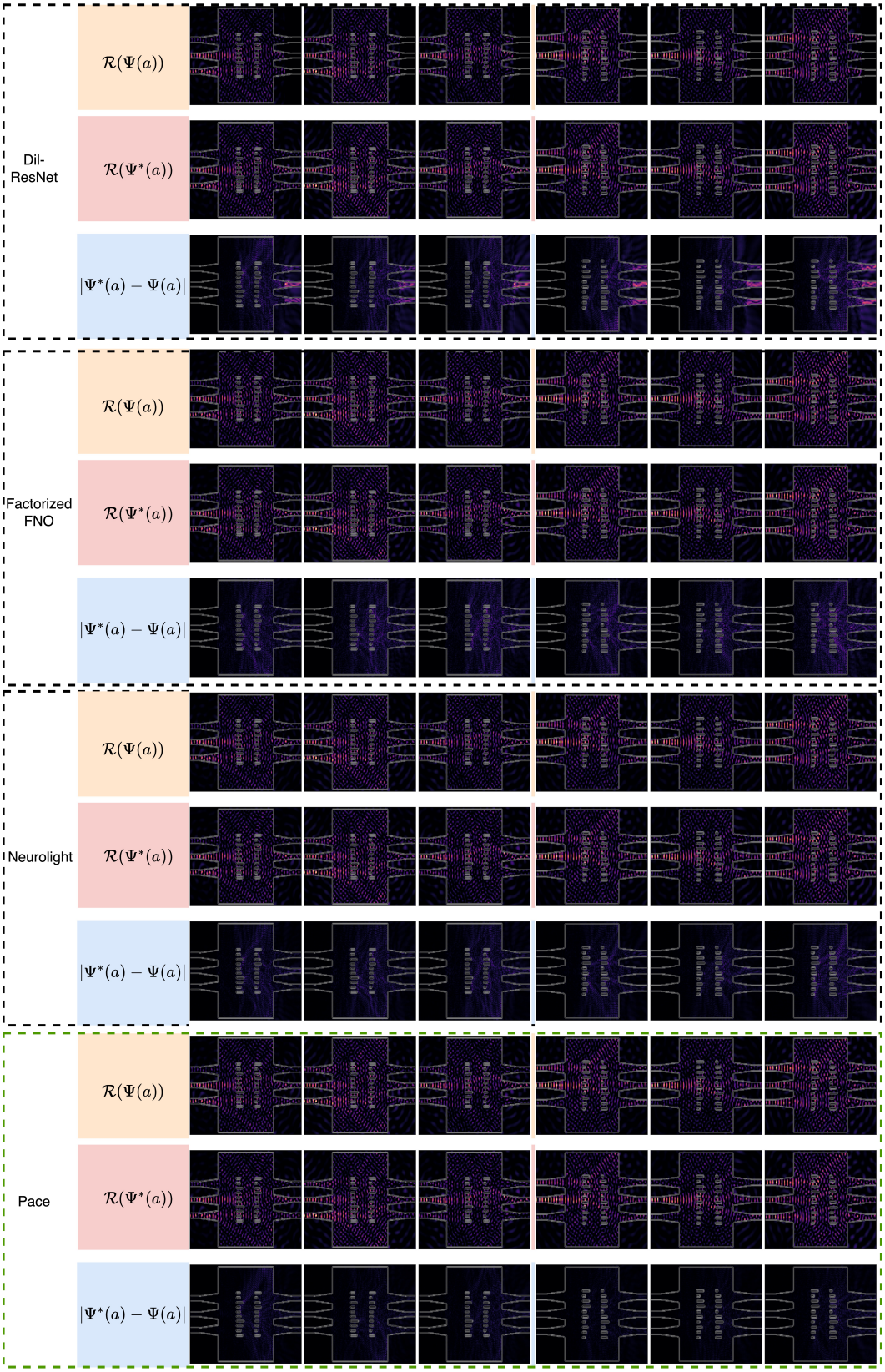



Figure 13: Visualization of several test cases on Metaline devices with random input sources.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer:[Yes]

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer:[Yes]

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer:[Yes]

   Justification: We open-source the dataset and code.

   Guidelines:

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer:[Yes]

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer:[Yes]

   Justification: We report the error bar when we test the performance on generalization experiments.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer:[Yes]

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer:[Yes]

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer:[Yes]

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer:[NA]

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer:[NA]

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer:[Yes]

    Guidelines:

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:[NA]

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer:[NA]